ISSN 1345-0093

愛知大学情報メディアセンター紀要

Vol.32/No.1 2023.March

COM

情報メディアセンター利用案内

◇サービス時間〈月～土曜日〉（都合により変更する場合があります。掲示およびホームページをご覧下さい。）

豊橋校舎

区　　分	曜日	420教室 （オープンアクセスルーム）	メディアゾーン （図書館）　※1	413教室・421教室・ 423教室・523教室	SEサービス
講義期間	月～金	9：10～19：00	9：00～19：50	講義利用のみ※2	9：00～17：40
講義期間以外	月～金	9：10～17：00	9：00～17：50		
通年	土	9：10～13：00	10：00～16：50		なし

※1　メディアゾーンは，豊橋図書館の運用日程に準じます。
※2　講義期間の木曜と土曜，及び，講義期間以外の期間は全て，413教室を開放教室として使用します。

名古屋校舎

区分	厚生棟4F　※3 メディアゾーン受付	講義棟7F メディアカウンター	SEサービス
講義期間	（月～金） 8：50～21：00 （土） 8：50～17：00	（月～金） 8：50～20：00 ※土曜日は閉室	（月～金） 8：50～18：30
講義期間以外	（月～土） 8：50～17：00	閉室	（月～金） 8：50～17：00

※3　メディアゾーンは，名古屋図書館の運用日程に準じます。

車道校舎

区分	メディアゾーン	K802，K804	SEサービス
講義期間	（月～金） 9：00～21：00	講義利用のみ	（月～金） 9：00～17：00
講義期間以外	（月～金） 9：00～18：00		

■センター閉室日　／　日曜・祝日・夏期休暇期間（8月土曜日）・年末年始・創立記念日（11/15）・
入試期間ただし，祝日授業日は開室

はじめに

情報メディアセンター長　岩田　員典

本年度も愛知大学情報メディアセンター紀要 COM47 号を無事発刊できました。お忙しい中投稿していただいた執筆者の方々をはじめ，編集委員や発刊に関わっていただいた方々にお礼を申し上げます。今号は大学院生からの投稿も含め全 4 件となっております。多くの方々にご愛読いただければ幸いです。

さて，昨年末から米国 IT 企業における大量解雇がニュース等で大きく報じられています。メタは 1 万 1000 人以上、Amazon は 1 万人の人員削減を発表しました。また，Google も 1 万人を解雇する予定のようです。さらに，Twitter はイーロンマスク氏による買収後に社員の半数に当たる約 3,700 人を解雇しました。これらの大手 IT 企業などは新型コロナウイルスが広まった 2020 年から「巣ごもり需要」で大きく業績を伸ばし，それに伴い多くの人を雇うだけでなく，人員の確保のため賃金を上げたことが影響しているようです。また，各企業がネット広告への費用を絞り込んだことも関係していると考えられます。経営上の判断とは言え現時点でのこのような大量解雇は，将来の IT 分野の成長を阻害する恐れがあると言われています。一方，日本の IT 業界はどうでしょうか？先日，IT 企業の方にお話を伺う機会がありました。そこで聞いたのは，とにかく IT 関連の人材が不足しており，文系出身でも IT 関連に興味があれば是非就職先として検討して欲しいとのことでした。本学は学部としては文系しかないですが，共通教育科目などを通じて IT について学ぶ機会はいろいろと用意されています。これらの授業と情報メディアセンターの活動により，これからの時代に必要とされる人材を育てていければ思っております。

最後になりましたが，早いもので，2022 年 9 月 30 日を以て情報メディアセンター所長の第 2 期目の任期を終えることができました。1 期目の途中から新型コロナウイルスが始まり，あっという間に 2 期 4 年が経過したように感じます。その間至らぬ点も多かったと思いますが，皆様のおかげでなんとか乗り切ることが出来ました。通例では 2 期で交代となることが多いようですが，未だに続いている新型コロナウイルスのこともあり，力不足ながら引き続き情報メディアセンター所長を務めさせていただくことになりました。今後も事務スタッフやICT 委員の皆さんの御協力を得ながら努めて参りたいと思っております。よろしくお願いいたします。

---

[1] ただし，IT 関連企業の労働条件（特にアメリカなどに比べて低い賃金）が改善されていかないと，人材の確保は難しいといえる。

目　　次

Unity ML-Agents 実行環境の構築と強化学習における報酬の影響の検討

岩田　員典（経営学部）
勝又　洋斗（経営学部）

要旨

近年，Artificial Intelligence（人工知能，以下 AI）の発展はめざましく，さまざまな分野に浸透してきている。特に機械学習の一種であるディープラーニングを利用したさまざまなアプリケーションは著しい成果を上げている。しかし，機械学習は初期値やパラメータにより学習結果が大きく変わる。そこで，本論文では機械学習の一つである強化学習において報酬がどのような影響を及ぼすかについて調査する。そのために，Unity が提供する Unity Machine Learning Agents（以下，ML-Agents）を使用する。しかし，ML-Agents を実行する環境を構築するのは，頻繁にアップデートされることもあり煩雑である。そこで，本論文では Windows，Linux，Mac における ML-Agents の実行環境の構築方法について説明をする。そして，この環境を使って，サンプルに含まれる人型エージェント「Walker」の報酬を変更した場合の学習結果について検証をする。

キーワード：Unity ML-Agents，機械学習，強化学習

1. はじめに

AI という言葉は今や，我々の日常生活の中で見かけることが珍しくないほど頻繁に使用されている。特に機械学習の一種であるディープラーニング[1]を利用したさまざまなアプリケーションは著しい成果を上げている。例えば，テーブルゲーム AI は急速に進化しており，完全情報ゲームであるオセロ[2]，チェス[3]，将棋[4]，碁[5]などでは AI はプロに勝利するほどに成長している。更に，プレイヤー同士の読みあいなどが発生する不完全情報ゲームである，大富豪（大貧民）[6]，ポーカー[7]や人狼ゲーム[8]などのテーブルゲームでも AI の研究が進められている。

このように様々な分野で活用されている機械学習ではあるが，初期値やパラメータにより学習結果が大きく変わる。そこで本論文では，機械学習を行う際にパラメータがどのような影響を及ぼすかについて調査する。そのため，Unity が提供する Unity ML-Agents[9]を使用し，サンプルに含まれている学習環境「Walker」の報酬を変更し，その影響について検証する。また，ML-Agents の実行環境の構築は，OS によって要求されるライブラリのバージョンが異なるなど

煩雑である。そこで，Windows，Linux，Mac における ML-Agents の環境構築方法と実行方法について説明する。

本論文の構成は次の通りである。まず，第 2 章で AI と機械学習について説明をする。次に，第 3 章で Unity ML-Agents と機械学習について述べる。そして，第 4 章で Unity ML-Agents の学習環境の構築方法について，第 5 章で学習の実行と結果の利用方法について説明をする。第 6 章でこれらを活用した報酬が与える影響と学習結果について論じる。最後に第 7 章でまとめと今後の課題について述べる。

2. AI と機械学習

本章では AI について簡単に説明をし，機械学習やディープラーニングの位置づけについて述べる。

2.1. AI の定義と現状

AI とは，「人工的にコンピュータ上などで人間と同様の知能を実現したもの」と定義されている。しかし，一般的に非常に広い概念をもった言葉で，専門家によっても定義が異なる[10]。

AI にはレベルというものがあり，一般に「汎用人工知能」と「特化型人工知能」に大別される。汎用人工知能は，人間の知能に迫って人間の仕事をこなせるようになり，幅広い知識と何らかの自意識を持つとされる。「人工知能によって人類が滅ぼされる，人間の知能を人工知能が上回る」と言った文脈で使われるものは汎用人工知能に分類できる。

一方，特化型人工知能は，全認知能力を必要としない程度の問題解決や推論を行うソフトウェアを指す。例えば，前述した囲碁のプロ棋士に勝利した AlphaGo[5] や，ディープラーニングなども，この特化型人工知能に区別される。

2022 年現在，汎用人工知能は実現されておらず，現存しているのは特化型人工知能のみである。

2.2. 機械学習

機械学習とは大量のデータの中から規則性を見つけ，推論に有用な規則，ルール，判断基準などを機械に生成させる手法である。AI の研究分野の 1 つである。

機械学習が用いられる以前の AI は，予測や判断を行うためのルールを全て人間が考える必要があった。よって AI の限界は，人間の限界によるものとなっていた。

しかし，機械学習の登場により，ある特定の分野においては，AI が人間の知能を超えることができるようになってきている。

2.2.1.「学習」と「推論」

機械学習には「学習」と「推論」という２つのプロセスが存在する。

「学習」とは，大量の学習データの統計的分布から特徴を抽出し，実際のデータから推論するための特徴の組み合わせパターン（推論モデル）を生成するためのプロセスである。

「推論」とは，分類や識別をしたいデータを，「学習」で生成した特徴の組み合わせパターン（推論モデル）に当てはめ，推論結果を導き出すプロセスである。

2.2.2. 機械学習の手法

機械学習の手法にはいくつかの種類がある。代表的なものとして「教師あり学習」，「教師なし学習」，「強化学習」などがある。

教師あり学習

教師あり学習（Supervised Learning）は，学習データに正解ラベルを付けて学習する手法である。予測の基礎となる正解ラベルと，学習に使用する学習データをセットで学習させることにより，入力されたデータに対して予測を出力する推論モデルを生成する。

教師あり学習は大きく「分類」と「回帰」の２つに分けることができる。

「分類」とは入力されたデータに対し，出力としてデータの属性または種類を返す手法である。迷惑メールかどうかを判断するスパムフィルタなどがこれに該当する。

「回帰」とは入力されたデータに対し，出力として数値を返す手法である。商品の販売予測などがこれに該当する。

教師なし学習

教師なし学習（Unsupervised Learning）は，データの構造を学習させる手法である。

学習データのみで学習させ，そのデータに含まれた潜在的なパターンを見つけ出す推論モデルを生成する。最も一般的な教師なし学習の手法は「クラスタリング」である。

「クラスタリング」は学習データのパターンを見つけ出し，似たパターンを持つ性質の近いデータ同士をまとめる手法である。類似購買者のグルーピングなどがこれに該当する。

強化学習

強化学習（Reinforcement Learning）は，ある「環境」に置かれた「エージェント」が環境から「状態」に関する情報を得る。そして，その「状態」に対し「行動」し，得られる「報酬」が最大化するような方策を求め，推論モデルを生成する手法である。「Unity ML-Agents」において，主に使用される学習方法である。他にも，将棋や囲碁などの対戦ゲーム AI や，車の自動運転などに利用される（図 1）。

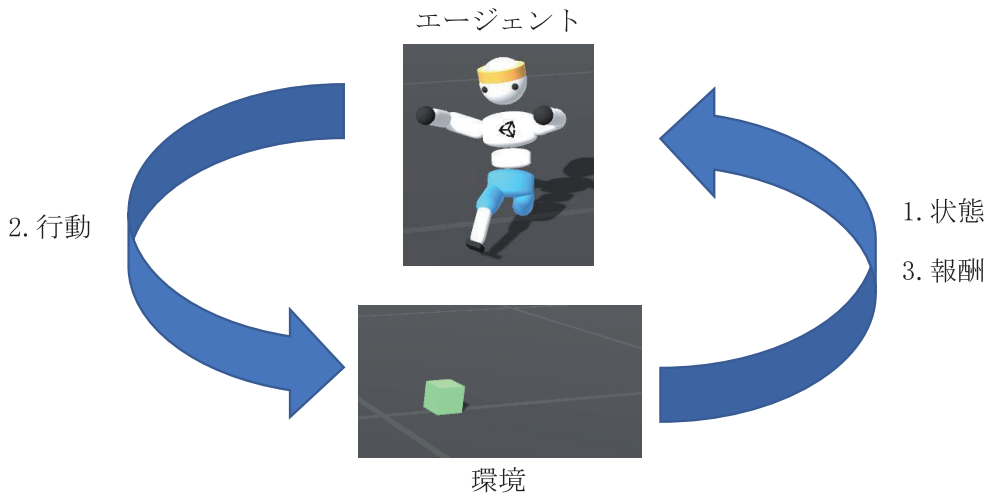


図１　強化学習の例

2.3. ディープラーニング

ディープラーニング[1)]は，教師あり学習の１つである「ニューラルネットワーク」を元に開発されたものである。本節ではニューラルネットワークとディープラーニングについて概要の説明をする。

2.3.1. ニューラルネットワーク

ニューラルネットワークとは，人間の脳の神経細胞を模した数理モデルである。ニューラルネットワークは「入力層」,「中間層」,「出力層」を持つ階層構造で構成される（図 2)。

各層にはノード（図２中丸印）がいくつも配置され，それぞれのノードの間はエッジ（図２中矢印）で結ばれる。ニューラルネットワークでは入力層から中間層，中間層から出力層へとデータを伝播させていく。各層のノードには数値データが格納されている。異なる層間のノード同士はエッジで結合しており，ノードに格納された数値データは，エッジを介して次のノードへと伝播していく。このとき，エッジには「重み」が付与されており，データが入力層から伝播して出力層へたどり着くと出力値（予測値）を得ることができる。ニューラルネットワークにおいて一般的に,「重み」は特定の個体ごとに値を設定され,「重み」はシナプス結合の強さを表す。中間層では，予測の精度を高めるための学習が行われる。

2.3.2. ディープラーニング

ディープラーニングは，ニューラルネットワークの多数の中間層があるモデルを構築できるため,「ディープニューラルネットワーク」(Deep Neural Network：DNN) とも呼ばれる。従来のニューラルネットワークでは中間層は２～３層程度であっ

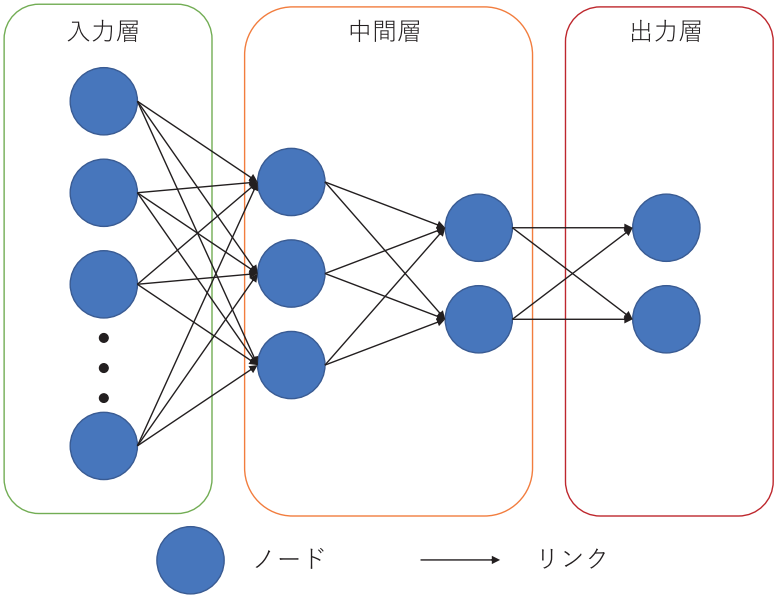


図２　ニューラルネットワークの基本構成

たが，ディープニューラルネットワークでは中間層は 100 を超える学習も可能になっている。こういった多層での学習が可能となった要因として，コンピュータ処理能力，特に CPU（Central Processing Unit）や GPU（Graphics Processing Unit：画像処理に特化したプロセッサのこと）の性能向上などがあげられる。

また，画像認識などにおいては畳み込みニューラルネットワーク（Convolutional Neural Network，以下 CNN）が用いられている。CNN では畳み込み層とプーリング層を有している。これらの層を使うことで CNN は画像から特徴を抽出し，画像内における対象の位置にかかわらず高い精度での認識が行える。

このように，ディープラーニングは特に画像や音声，言語などの非構造化データから特徴量を抽出し，精度の高い推論モデルを構築することに長けている。これらの非構造化データは説明変数の数が多く，特徴量の抽出が難しい。そのため従来の機械学習の手法を用いて精度の高い推論モデルを構築するためには，画像データや音声データ，自然言語処理の専門知識が必要である。一方，ディープラーニングを用いることにより，機械が自動的に特徴量を抽出するため，専門知識が得られないような場合にも精度の高い推論モデルを構築するが可能になっている。

3. Unity ML-Agents と機械学習

Unity[11] とはゲーム制作の大衆化を目

指して作られたゲームエンジンである。Unity を使用すれば 3D ゲーム開発や，物理エンジンなど，専門知識が少なくても導入することが可能である。

また，Unity ML-Agents は機械学習のためのライブラリ「PyTorch」[注1)] と Unity で作成した学習環境を連携させ，Unity のシーン内のキャラクターを学習させることが可能である。これを利用することにより，深い専門知識なしで，視覚的にわかりやすく機械学習を体験することができる。

Unity ML-Agents で利用可能な主な学習方法として「強化学習」「模倣学習」「強化学習＋模倣学習」がある。

3.1. 強化学習

強化学習は前述のように，ある「環境」に置かれた「エージェント」が環境に対し「行動」し，得られる「報酬」が最大化するような方策を求め，推論モデルを生成する手法である。学習に時間がかかるが，人間を超える能力を得られることがある。

3.2. 模倣学習

模倣学習は他人が行う行動を模倣して学習させる手法である。Unity ML-Agents では，人間（Player）が操作するエージェントの行動を模倣して学習する。そのため人間に似た能力を素早く得ることができる。

3.3. 強化学習＋模倣学習

強化学習＋模倣学習は，まずは模倣学習により「人間の行動」という「正解データ」を元に学習をする。そして，その学習結果を更に強化学習により改善をする。

3.4. その他の学習方法

その他にも，Recurrent Neural Network（RNN），Long Short-Term Memory（LSTM），カリキュラム学習等を利用することができる。

4. Unity ML-Agents の学習環境の構築

Unity ML-Agents を動作させるためには以下のソフトウェアを導入する必要がある。ただし，OS の種類によって動作するバージョンなどが異なるため各節で Windows，Linux，Mac の設定について説明する[注2)]。

- Unity Hub[12)]

注 1) 以前は TensorFlow が用いられていた。

注 2) 本論文執筆の 2022 年 11 月時点の情報

- Unity[12)]
- Unity-ML Agents[13, 14)]
- Python
- PyTorch[15)]
- CUDA[16)]・NVIDIA cuDNN[17)]（利用したい場合）

なお，Unity のインストールの際にユーザ登録やライセンスの取得が必要になる。大学教員の場合に，個人で使用する PC では Unity Educator Plan[18)] を使い，教室や研究室では教育機関向けライセンス[19)] を利用する必要がある。学生の場合は Unity Student プラン[20)] を利用するとよい。

4.1. Windows での設定

Windows10・11 では以下のバージョンの各ソフトウェアをインストールすることで，比較的容易に実行することができる。

- Unity Hub：3.3.0
- Unity：2021.3.21
- Unity ML-Agents：Release 19
- Python: Anaconda[21)] Python 3.9・64-Bit Graphical Installer を利用
- PyTorch：1.13

4.1.1. Unity Hub と Unity のインストール

Unity Hub をダウンロード（図 3）・インストールし，Unity Hub から Unity をインストールする。インストールは「初めて Unity をインストールする手順について＜ Windows 編＞」[22)] が参考になる。

Unity Hub の Preferences（図 4）から Appearance を選び Language を「日本語」にすることで，Unity Hub を日本語化できる（図 5）。

図 3　Unity Hub のダウンロード

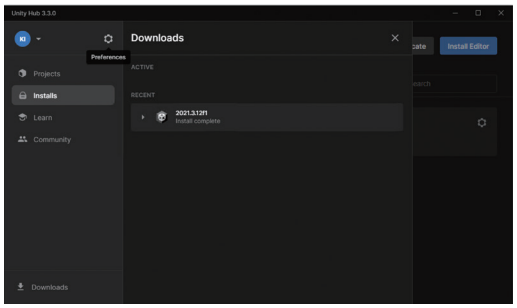


図 4　Unity Hub の Preference

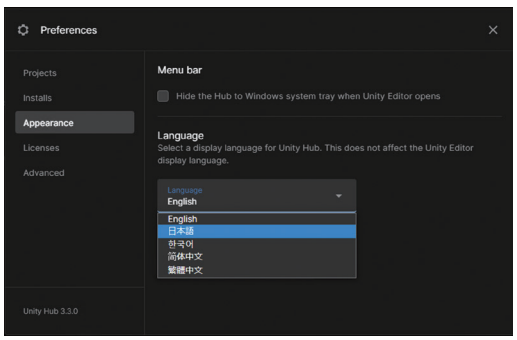


図 5　Unity Hub の言語の選択

Version	Release Date	Source	Documentation	Download	Python Package	Unity Package
main (unstable)	--	source	docs	download	--	--
Release 19	January 14, 2022	source	docs	download	0.28.0	2.2.1
Verified Package 1.0.8	May 26, 2021	source	docs	download	0.16.1	1.0.8

図 6　ML-Agents のバージョンとダウンロード

4.1.2. Unity ML-Agents のダウンロードと設定

利用したい Unity ML-Agents のバージョンを選びダウンロードする（図 6）。download をクリックすると zip ファイルが得られる。git を使っている場合は HTTPS 経由でも同様のファイルが得られる。

そして，上記のいずれかの方法でダウンロードした ml-agents-release_19.zip を展開し，任意の場所に移動させる。

次に，図 7 のように Unity Hub の「開く」から「ディスクから加える」を選び，ML-Agents のプロジェクトを加える。

加えるプロジェクトは「ml-agents¥Project」となる。ただし，ml-agents は展開したフォルダ名のため，「ml-agents-release_19」のような名称となることもある。

図 7　Unity へプロジェクトの追加

4.1.3. Python と PyTorch の準備

Python は Anaconda 経由でインストールする。

Anaconda と PyTorch のインストール方法は「コンピューターに PyTorch をインストールして構成する」[23] が参考になる。

また，NVIDA 製のグラフィックスボードを搭載している場合は CUDA をインストールすることで利用できる。

Anaconda Prompt を起動し以下のコマンドで Python の仮想環境を作成する。仮想環境の作成は必須では無いが，Anaconda の base 環境で pip3 を使うと Anaconda の動作がおかしくなることがあるため，こだわりがない限りは仮想環境を作成した方がよい。なお，仮想環境を作成する前に pip を最新にしておく。

```
> python.exe -m pip install
  --upgrade pip
```

ここで「>」はプロンプトを表している。またコマンドは一行だが誌面の都合上折り返して記載する（以下同様）。

また，Python のバージョン情報が必要なため，python--version で確認する。このバージョン（本論文執筆時では 3.9.13）を使い，以下のコマンドにより仮想環境を作成することができる。

```
> conda create --name ml-agents19
  python=3.9.13
```

仮想環境の作成ができたら以下のコマンドにより有効にする。

```
> conda activate ml-agents19
```

また，仮想環境を無効にし，元の環境に戻る場合は「conda deactivate」を実行する。なお，作成した仮想環境を確認する場合は「conda info -e」である。

PyTorch のインストールは CPU と CUDA によって異なるので注意が必要である。PyTorch のページ[15]から環境を選択すると下部に実行すべきコマンド Run this command：に表示される（図 8）ので，それをコピーして実行する。

PyTorch のインストール（CPU）

```
> conda install pytorch torchvision
  torchaudio cpuonly
  -c pytorch
```

PyTorch のインストール（CUDA）

```
> conda install pytorch torchvision
  torchaudio pytorch-cuda=11.7 -c
  pytorch -c nvidia
```

PyTorch がインストールできているかは，Python を起動し「import torch」を実行することで確認することができる。また，CUDA が利用できるようになっているかどうかは，「import torch」の後に「torch.cuda.is_available()」を実行することで確認が行える。

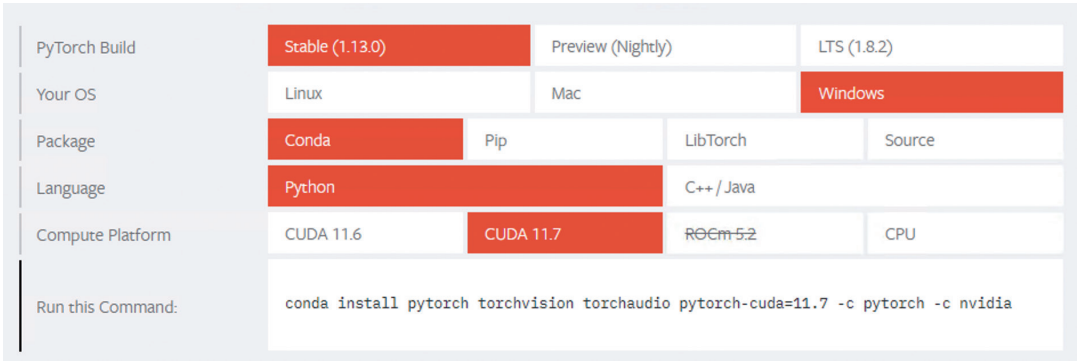


図 8　PyTorch の選択

```
> python
>>> import torch
>>> torch.cuda.is_available()
True
```

「True」と表示されればCUDAが利用できている。ここで「>>>」はPythonインタプリタの対話モードであることを表している。なお，対話モードから抜けるには「quit（）」と入力する。

4.1.4. Pythonによる ML-Agentsの設定

Anaconda Promptから，Downloadし展開したml-agentsのフォルダに移動する。例えば，Documentsの下に展開してあれば以下のようなコマンドになる。

```
> cd C:¥Users¥ユーザ名
  ¥Documents¥ml-agents-release_19
```

このフォルダ（ここではml-agents-release_19）以下にあるml-agentsとml-agents-envsの内容を編集可能な状態のパッケージとしてインストールする。

これにより，これらのフォルダ以下のファイルを編集しても，再インストールすることなしに編集結果を実行に反映できる。実行するコマンドは以下の通りである。

```
> pip3 install -e .¥ml-agents-envs¥
> pip3 install -e .¥ml-agents¥
```

上記の設定を行うとmlagents-envsとmlagentsの0.28.0がインストールされるが，python 3.9.13では0.29.0に変更しないとエラーが起きる。そこで以下のコマンドを実行する。なお，mlagents-envsのインストールで警告が出るが無視してよい。

```
> pip3 install mlagents-envs==0.29.0
> pip3 install mlagents==0.29.0
```

4.2. Linux（Ubuntu）での設定

本節ではUbuntu 22.04.1 LTSでML-Agentsを利用するための設定について説明をする。Ubuntuでは以下のバージョンの各ソフトウェアを使う。

- Unity Hub：2.4.6
- Unity：2022.1.23f1[注3)]
- Unity ML-Agents：Release 19
- Python：Anaconda Python 3.7・64-Bit Graphical Installerを利用
- PyTorch：1.8.2

注3) 本論文執筆時には2021.3.14f1(LTS)が推奨されるが，インストール時にエラーになる。また，いずれのバージョンを利用してもProjectがうまく読み込めないという問題が生じている。

4.2.1. Unity Hub と Unity のインストールと ML-Agents の設定

「Unity をダウンロード」[24] から「Unity Hub をダウンロード」（図 9）を選び，「UnityHub.AppImage」をダウンロードする（本来ならば Installing the Hub on Linux[25] に従って Unity Hub をインストールするべきだが，起動時にエラーが生じる）。

図 9　Unity Hub のダウンロード

ダウンロードができたら「UnityHub.AppImage」に実行権限を与える。Google Chrome からダウンロードした場合は，ホームディレクトリの下の Downloads に保存されているはずなので，次のようなコマンドを実行する。

```
> cd Downloads
> bash Anaconda3-YYYY.MM-Linux-x86_64.sh
```

また，fuse がインストールされている必要があるため，未インストールの場合は apt を使いインストールしておく。

```
> sudo apt install fuse libfuse2
> cd Downloads
> chmod 777 UnityHub.AppImage
```

実行権限を付与しても，そのまま実行するとエラーが生じるので，以下のように --no-sandbox オプションを付けて実行する。

```
> ./UnityHub.AppImage --no-sandbox
```

UnityHub が起動したら 4.1.1・4.1.2 と同様に Unity や Unity ML-Agents の設定を行う。

4.2.2. Python と PyTorch の準備

Windows と同様に Python は Anaconda 経由で設定をする。Anaconda のインストールは「【Ubuntu Server 18.04】Anaconda をインストールする」[26] が参考になる。

Anaconda は「ANACONDA DISTIRUTION」[21] にアクセスし，Download をクリックする。Anaconda3-YYYY.MM-Linux-x86_64.sh をダウンロードすることができるはずなので，以下のようにこのファイルを実行する（YYYYY には西暦が，MM には月が入る）。

ライセンスの認証には「yes」と入力し，インストール先はデフォルトの「/home/<USERNAME>/anaconda3」を選択する。

これらの設定が完了したら，利用する python を Anaconda のものにするために，ターミナルを再起動するなど，シェルの設定を読み込み直す。以下のように python の場所を調べて，上記のインストール先の bin 以下にある python が指定されていればよい。

```
> which python
/home/<USERNAME>/anaconda3/
bin/python
```

その後は，4.1.3 と同様に設定を完了させる。ただし，Ubuntu 22.04.1 LTS 上では python のバージョンが 3.7 でないとエラーになるため，下記のコマンドを使い設定を行う。

- pip のアップグレードと ML-Agents のダウンロード

```
> python --m pip install --upgrade
  pip
> git clone
  https://github.com/Unity-
  Technologies/ml-agents.git
```

- 仮想環境の設定と PyTorch のインストール

```
> cd ml-agents
> conda create --name ml-agents19
  python=3.7
> conda activate ml-agents19
> conda install pytorch torchvision
  torchaudio cpuonly -c pytorch-lts
```

4.2.3. Python による ML-Agents の設定

ML-Agents の設定も 4.1.4 と同じように行うことができる。前項のコマンドを実行している場合は ml-agents ディレクトリに移動済みのはずなので，以下のコマンドを実行することで設定が完了する。

```
> pip install -e ./ml-agents-envs/
> pip install -e ./ml-agents/
> pip install mlagents-envs==0.29.0
> pip install mlagents==0.29.0
> pip install importlib-
  metadata==4.4
```

なお，これらの実行が完了したら，パスの設定を更新するために再度ターミナルを再起動する必要がある[注4)]。

4.3. Mac での設定

本節での説明は Intel プロセッサと Apple シリコンの両方に対応している。また，macOS Ventura 13.0.1 が対象である。

利用するソフトウェアは以下の通りである。ただし，執筆時点で ML-Agents に対応する PyTorch が Mac では動作しないため，Python や PyTorch は Docker[27)] 上で動作させる。この Docker の設定についても後述する。

- Unity Hub：3.3.0

注 4) rehash などを使ってもよい

- Unity：2021.3.13f1
- Unity ML-Agents：Release 19
- Docker v4.13.1

  Docker 内部の設定
  - Ubuntu 22.04
  - Python: 3.7.1
  - PyTorch: 1.8.1

4.3.1. Unity Hub と Unity のインストール

「Unity をダウンロード」[24]から「Unity Hub をダウンロード」をクリックすると「UnityHubSetup.dmg」がダウンロードされるはずなので，ダブルクリックしインストールをする。インストールが完了したら，アプリケーションフォルダから UnityHub を起動し 4.1.1 と同様に設定を進める。ここで，ML-Agents は Docker からダウンロードするため 4.1.2 の設定はまだ行わない。

4.3.2. Docker を利用した Python と PyTorch の準備

前述のように ML-Agents に対応する PyTorch が Mac 向けには提供が終了しているため，Docker を利用して ML-Agents を動かせるようにする。Docker 内部で起動させた OS は Ubuntu 22.04 のため，基本的な設定は 4.1.3・4.1.4 と同様である。また，Dockerfile や各種スクリプトは GitHub に「macOS で Unity ML-Agents19 を動作させるための Docker の設定」[28]として公開している（付録 1 ～ 5）。

Agent のスクリプトなどを編集しやすくするために ml-agetns は Mac 上に置き，バインドマウントで Docker にマウントを行う方式を採用している（図 10）。

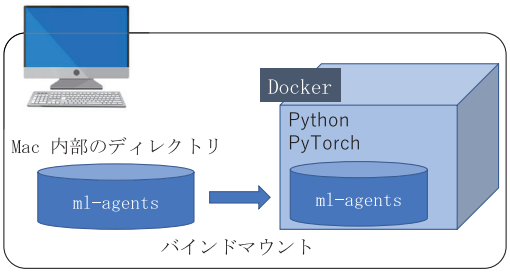


図 10　ml-agents のマウント

ml-agents を置きたいディレクトリに移動し，以下のコマンドを実施することでダウンロードから build まで行うことができる。ただし，あらかじめ docker や git が使えるように設定がなされている必要がある。

```
> git clone https://github.com/kazunori-iwata/ml-agents19-for-macOS.git
> sh ./docker-build.sh
```

4.3.3. Unity ML-Agents の設定

Docker 上で ML-Agents を動作させるには，Unity から Linux 対応のビルドを実施する必要がある。少し情報が古いが「Docker で Unity ML-Agents を動作させてみた（v0.11.0 対応）」[29]の「Unity に Linux ビルドサポートコンポーネントを追加する」が参考になる。

図 11　Unity へのモジュールの追加

まず，以下の手順でモジュールを追加する。

1. Unity Hub の「インストール」から Unity の右側にある歯車をクリックして「モジュールを加える」を選択する（図 11）
2. 表示されたモジュールを加えるにある「Linux Build Support（IL2CPP）」にチェックを入れて「インストール」ボタンをクリックする（図 12）

次に，ML-Agents に含まれているサンプルを学習できるようにする。4.1.2 の図 7 と同様にプロジェクトの追加をする。その際に，4.3.2 で保存したディレクトリ以下を指定する必要がある。

また，本論文ではサンプルにある Walker を利用するため，プロジェクトが読み込めたら「Project」タブから「Asserts → ML-Agents → Examples → Walker → Scenes」にある「Walker」をダブルクリックする（図 13）。

Unity 上部に図 14 のようなエージェントが表示されれば Scene の設定は完了である。

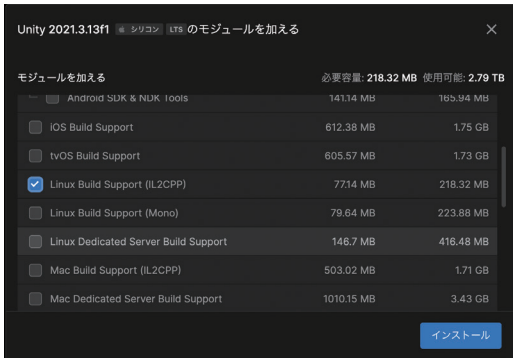


図 12　モジュールのインストール

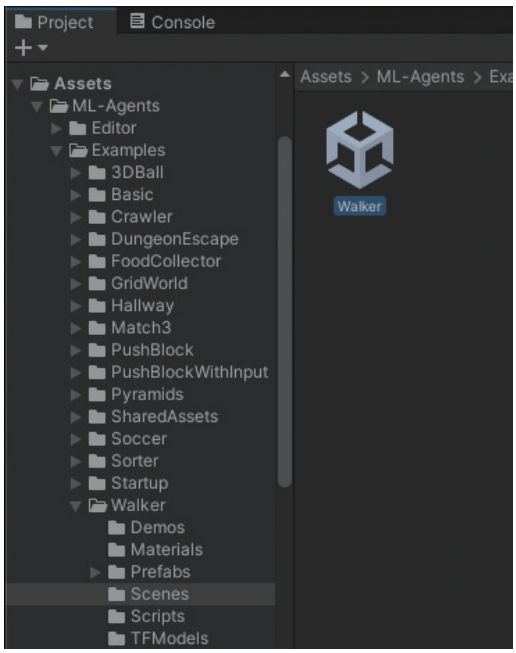


図 13　Scene を開く

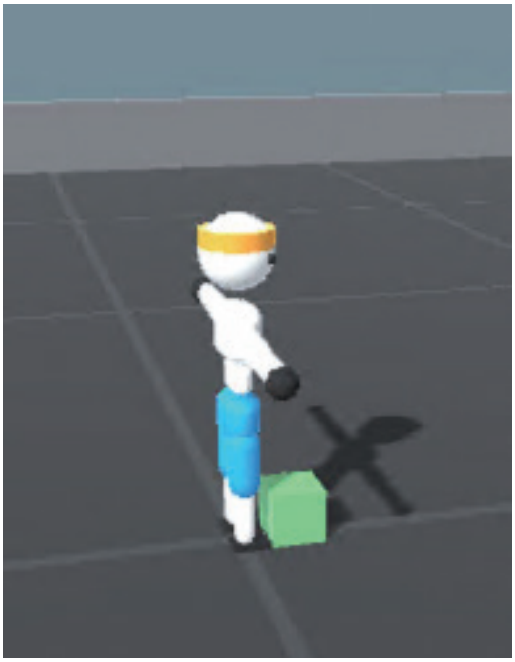


図 14　エージェントの設置例

Scene が設置できたら，ビルドの設定をし Linux で動作するようにする。

1. Unity の「File」メニューから「Build Settings...」を選ぶ（図 15）
2. 「Target Platform」から「Linux」を選択する。
3. 「Add Open Scenes」をクリックすると「ML-Agents/Examples/Walker/Scenes/Walker」が追加されるので，Scenes In Build にチェックマークが入っていることを確認する（図 16）。
4. 「Build」をクリックし（図 16），保存先を指定する。保存先は ML-Agents のトップディレクトリである「ml-agents」の下の「unity-volume」にする。また，ファイル名は「Walker」とする（図 17）。

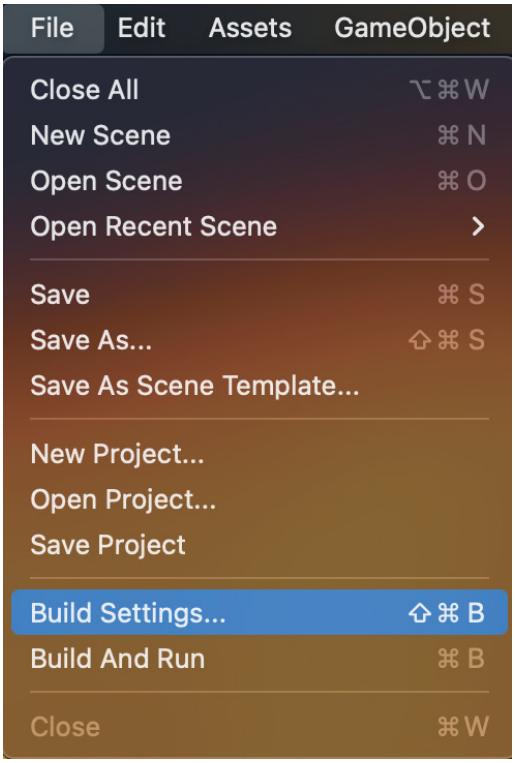


図 15　Build Settings

5. ML-Agents の学習の実行と結果の利用

ML-Agents では 17 種類のサンプル[30]が用意されており，さまざまな環境においてエージェントを学習させることができる。本論文では前述のように Walker を対象とする。

5.1. Walker の概要

Walker は以下のパーツに関して自由度 26 を有した人型ロボットである（図 18）。

- 腰
- 胸部

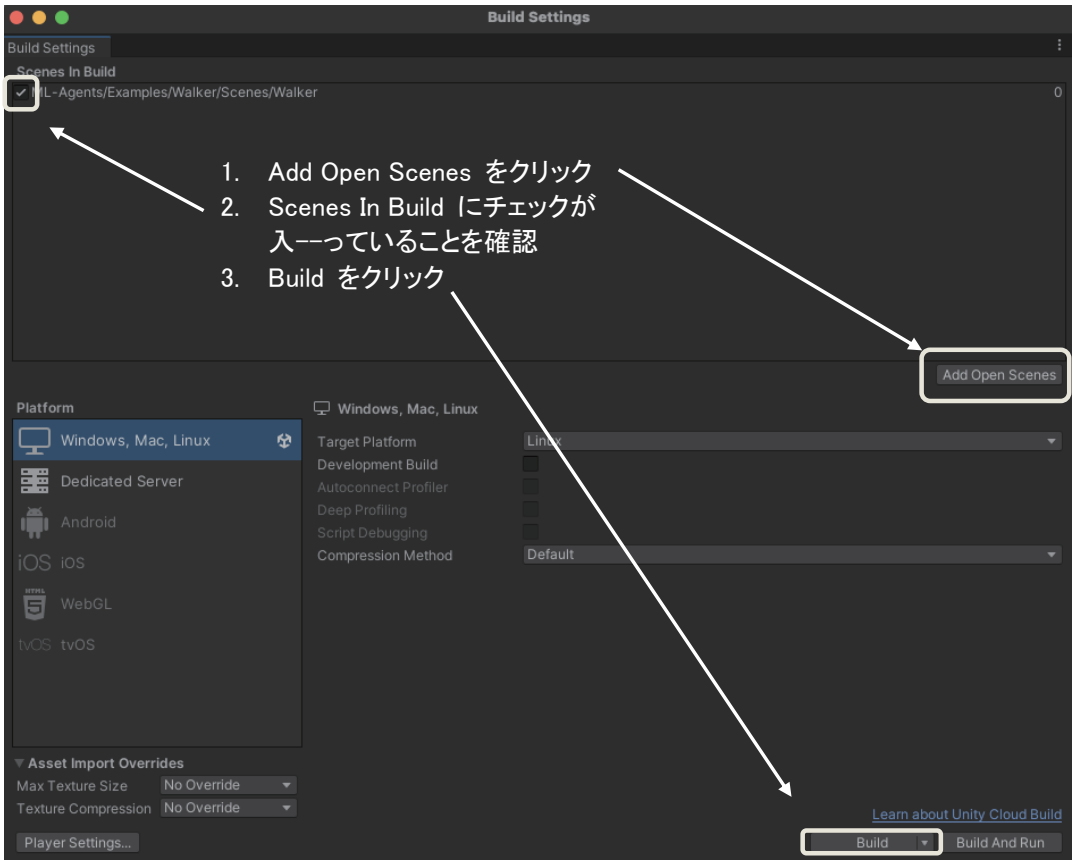


図 16　Build する Scene の設定

図 17　Build の保存先の指定

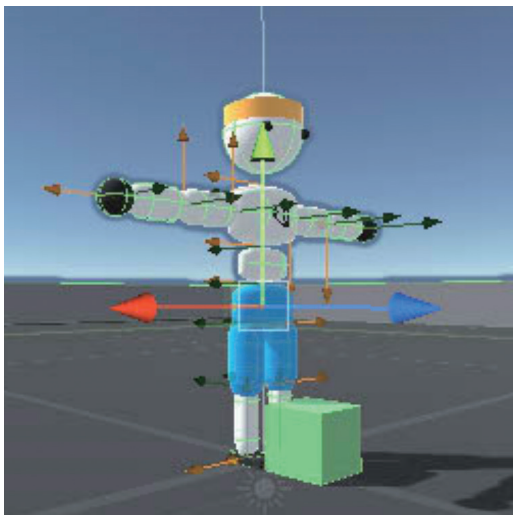


図 18　Walker エージェント

- 背骨
- 頭部
- 大腿部
- 脛骨部
- 足部
- 腕
- 前腕
- 手

このサンプルにおいてエージェントは，ランダムな位置に配置されるターゲット（ボックス）に到達することを目的とする。

5.2. 報酬（Rewards）の計算方法

強化学習の報酬は以下の方針に従って計算される。

- FixedUpdate のタイミングで報酬を受け取る。これは 0.02 秒に 1 回実施される。
  本来の強化学習ではエピソードが終了した際に合計値を受け取るが，学習を早くするためにこの方式が採用されている。
- 以下のいずれかの値になる。
  - ターゲットに到達したステップは 1
  - それ以外のステップは以下の 2 値の積
    - 全パーツの平均ベクトルがターゲットの方向にどの程度一致しているかを基準に判断する[注5)]。0 以上 1 以下の値に正規化されている。転んだ場合は 0 となる
    - 顔（頭部）がターゲットの方を向いているかどうかを基に計算される。0 以上 1 以下の値に正規化され，転んだ場合は 0 となる。

5.3. 学習の実行

学習は「mlagents-learn」（Windows では mlagents-learn.exe）によって行える。このプログラムはターミナルなどから実行する。Windows の場合は Anaconda Powershell Prompt で実行する必要がある[注6)]。

mlagents-learn の引数に学習の設定ファイルを指定し，オプション「--run-id」を使い学習結果を識別するための ID を指定する。この ID で指定された値を使って「ml-agents/results/<ID ＞」以下に実行結果が保存される。Walker を利用する場合に，学習の設定ファイルは「config/

注 5) 原文では「Body velocity matches goal velocity」と書かれているが，コードを読むと Body は bodyPartsList のベクトルの平均値を計算しており，goal velocity はボックスの方向に固定値 m_TargetWalkingSpee を乗算した値を利用している。

注 6) Anaconda Prompt では実行時にエラーになる。

ppo/Walker.yaml」である[注7)]。

また，「--torch-device=cuda:0」のように指定することでCUDAを利用することができる。

これらオプションについては「Unity ML-Agents Release 18 の mlagents-learn」[31)]で詳しく説明がされている。また，学習には時間がかかるためアプリ化して学習環境を複数起動することで高速化が可能である[32)]。

IDをfirstとした場合の実行方法は以下の通りである。

- Windows

```
> mlagents-learn.exe
  .¥config¥ppo¥Walker.yaml
  --run-id=first
```

- Linux

```
> mlagents-learn
  config/ppo/Walker.yaml
  --run-id=first
```

- Mac（Docker 利用）

```
> sh ./docker-run.sh
（以下 Docker 内）
> cd ml-agents
> mlagents-learn
  config/ppo/Walker.yaml
  --run-id=first
```

なお，コンテナを既に起動しており，exit している場合は，sh ./docker-attach.sh で再利用できる。

起動に成功すると図19のような表示の後に「Listening on port 5004. Start training by pressing the Play button in the Unity Editor.」と出力されるので，Unityの上部にある再生ボタンをクリックすることで学習が始まる。

図19　ml-agents の起動画面

学習が開始されると以下のように経過が表示される。

```
[INFO] Walker. Step: 30000. Time Elapsed: 12.147 s. Mean Reward: 0.247. Std of Reward: 1.623. Training.
```

ここで，Step が経過ステップ数，Mean Reward は1000ステップ毎の報酬の合計の平均値（この場合は30000ステップまでの1000ステップ毎），Std of Reward はその標準偏差を表す。

なお，本論文では以下の2台を利用し

注7)　本論文では Proximal Policy Optimization（PPO）を利用する

エージェントの学習を進めた。

- Ubuntu 22.04.1 LTS
  CPU: AMD Ryzen 7 5700G 3.8GHz
  RAM: 64.0 GB
- Mac Studio
  OS: macOS Ventura 13.0.1
  CPU: Apple M1 Ultra 3.2GHz
  20 コア CPU，64 コア GPU
  RAM: 128.0GB

5.4. 学習状況の監視・確認

学習がどのように進んだかは TensorBoard を利用することで確認することができる。

logdir オプションでは「results/<ID>」ディレクトリを指定する。ここでは，ID が first の場合について示す。

- Windows

```
> tensorboard.exe
  --logdir=.¥results¥first
```

- Linux

```
> tensorboard
  --logdir=./results/first
```

- Mac（Docker を利用）

```
> sh ./docker-attach.sh
（以下 Docker 内）
> cd ml-agents
> tensorboard
  --logdir=./results/first
  --host=0.0.0.0
```

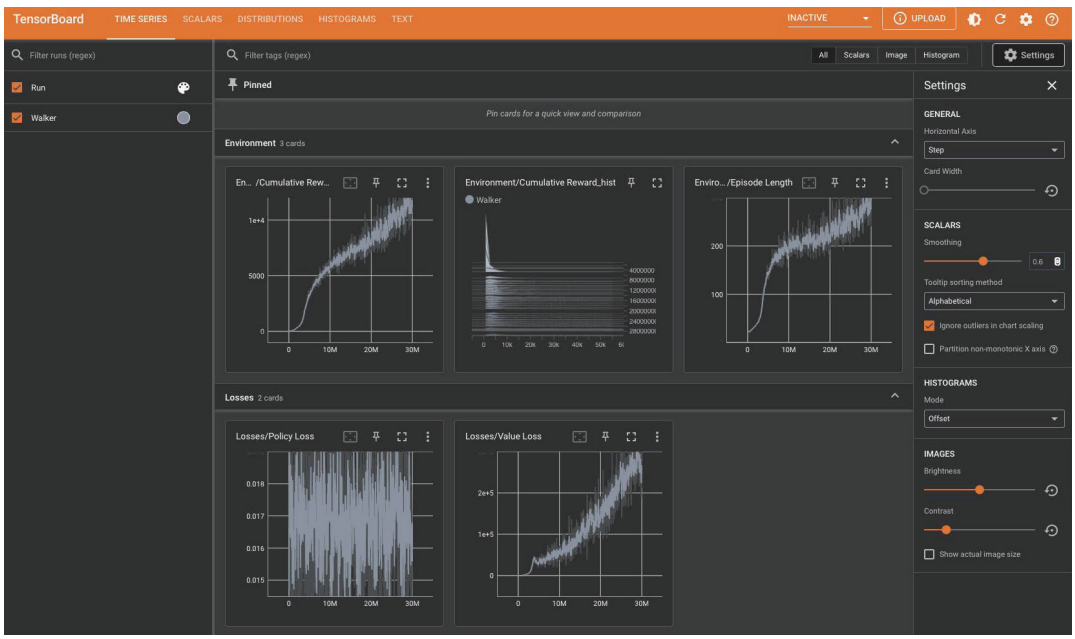


図 20　TensorBoard の例

これらのコマンドを実行して「TensorBoard 2.11.0 at http://localhost:6006/」のように表示されたらWebブラウザからhttp://localhost:6006/にアクセスすることで学習状況を監視したり確認したりすることができる（図20）。

5.5. 学習結果の利用

学習結果を利用するには以下の手順で設定をし，Unityの再生ボタンをクリックすればよい[33]。

1. results/<ID＞以下にあるWalker.onnxファイルをProject/Assets/ML-Agents/Examples/Walker/TFModels以下に置く。
2. UnityからPrefabsにあるBehavior ParametersのModelで配置したWalker.onnxファイルを指定しする
3. Behavior Typeをinferenceにする

しかし，本論文執筆時にはUnityからこれらを設定しても学習結果を利用できない[注8)]。そこで，Unity側ではなく，mlagents-learnを推論モードで実行することで，学習結果を利用する方法について説明をする。

mlagents-learnに「--inference」と「--resume」を与えることで推論モードとして起動することができる。--run-idには学習時に利用したIDを指定する。また，学習のTime Scale（表示速度）が初期値では20倍のため，--time-scale=1を指定する。

- Windows

```
> mlagents-learn.exe
  .¥config¥ppo¥Walker.yaml
  --run-id=first
  --inference --resume
  --time-scale=1
```

- Linux

```
> mlagents-learn
  config/ppo/Walker.yaml
  --run-id=first
  --inference --resume
  --time-scale=1
```

- Mac（Dockerを利用）

```
> sh ./docker-attach.sh
（学習でDockerのコンテは起動済みのはず。以下Docker内）
> cd ml-agents
> mlagents-learn
  config/ppo/Walker.yaml
  --run-id=first
  --inference --resume
  --time-scale=1
```

注8) 筆者の調査不足の可能性もあるため，確定情報ではない。また，参考文献33)の最新版が2022年12月に発刊が予定されており，その中で正確な手法が説明されているかもしれない。

6. 報酬が与える影響と学習結果

サンプルに用意されている設定で学習を進めるとWalkerはすり足でターゲットに接近をする。これは転倒すると報酬が0になるため，転倒しにくい移動方法を学習したからであると考えられる。

そこで，本論文では報酬を決定する方法を変更することで，すり足ではなく普通に人が移動しているような歩き方に近づけることを目差す。修正はProject/Assets/ML-Agents/Examples/Walker/Scriptsに置かれている「WalkerAgent.cs」に対して行う。

6.1.1. 情報取得部分の修正

姿勢の制御に関わる情報が不十分なため，一部修正する。姿勢の制御については「物理ベースのキャラを強化学習向けに調整する」にある「手順：既存のスクリプトを修正する」[34]を基に修正をする（付録6）。

6.1.2. 報酬部分の修正

報酬に関してはメソッド「FixedUpdate」で決定しているため，この部分を編集する。変更内容は以下の通りである（詳細は付録を参照）。

- 左右いずれかの足先が，逆の足よりも高い位置にあり，腰よりも低ければ高さに応じた報酬を与える。0以上1以下の値に正規化をする（付録7）。
- 頭の位置が腰の位置よりも高い場合に報酬を与える。こちらも0以上1以下の値に正規化をする（付録8）。

二つ目の報酬は設定しなくても足を上げて歩行を行うが，バランスをとろうと頭を大きく振ってしまう。そこで，頭の位置をあまり移動させないようにするため，この条件を加えている。

これらの学習の結果をYouTubeで公開している[35]（図21のリンク先）。

図21　学習の経過動画へのリンク（YouTube）

7. まとめと今後の課題

本論文では機械学習の一つである強化学習において報酬がどのような影響を及ぼすかについて検証した。この検証にはUnityが提供するML-Agentsを利用した。このML-Agentsの実行環境の構築方法や利用方法は複雑な上，OSによって異なるためWindows，Linux，Macの各OSでの方法について解説した。そして，この環境を使って，サンプル「Walker」の報酬を変更した場合の学習結果について

考察した。

今後は，本論文では動作が確認出来なかったUnityにおけるonnxの利用方法や，その他のエージェントの動作について検証を行う予定である。

参考文献

1）Yann LeCun, Yoshua Bengio, Geoffrey Hinton, “Deep Learning”. Nature. 521 (7553) : 436-444. Bibcode:2015Natur.521..436L. doi:10.1038/nature14539. PMID 26017442. S2CID 307409, (2015).

2）LOGISTELLO, 2002年11月更新. https://skatgame.net/mburo/log.html, ［アクセス日 : 2022年10月30日］.

3）Stockfish Outlasts “Rybkamura”, 2014年8月25日更新. https://www.chess.com/news/view/stockfish-outlasts-nakamura-3634, ［アクセス日 : 2022年10月30日］.

4）山本 一成, 下山 晃, 齋藤 真樹, 藤田 康博, 秋葉 拓哉, 土井 裕介, 菊池 悠太, 奥田 遼介, 須藤 武文, 大川 和仁, “第27回世界コンピュータ将棋選手権 Ponanza Chainer アピール文章”, 第27回世界コンピュータ将棋選手権, (2017)

5）Google Research Blog, “AlphaGo: Mastering the ancient game of Go with Machine Learning”, 2016年1月27日更新. https://ai.googleblog.com/2016/01/alphago-mastering-ancient-game-of-go.html, ［アクセス日 : 2022年10月31日］.

6）UECコンピュータ大貧民大会 2022（UECda-2022), 2022年9月31日 更 新. http://www.tnlab.inf.uec.ac.jp/daihinmin/2022/［アクセス日 : 2022年10月31日］.

7）Noam Brown, Tuomas Sandholm, “Superhuman AI for multiplayer poker”. Science. 365 (6456) : 885-890. doi:10.1126/science.aay2400, 2019.

8）人狼知能プロジェクト, 2022年10月4日更新. http://aiwolf.org/［アクセス日 : 2022年10月31日］.

9）Unity Machine Learning Agents, https://unity.com/ja/products/machine-learning-agents［アクセス日 : 2022年10月31日］.

10）人工知能学会, What’s AI, https://www.ai-gakkai.or.jp/whatsai/［アクセス日: 2022年10月31日］.

11）Unity, https://unity.com/ja［アクセス日 : 2022年10月31日］.

12）Unity download archive, https://unity3d.com/get-unity/download/archive?_ga=2.39544163.1505483174.1667441172-525719297.1667441172［アクセス日 : 2022年10月31日］

13）ML-Agents 2022年1月14更新. https://github.com/Unity-Technologies/ml-agents［アクセス日 : 2022年10月31日］.

14）Installing ML-Agents for Windows, 2018年3月5日 更 新. https://github.com/miyamotok0105/unity-ml-agents/blob/master/docs/Installation-Windows.md［アクセス日 : 2022年10月31日］.

15）PyTorch https://pytorch.org/［アクセス

日 : 2022 年 10 月 31 日].

16) Accelerated Computing Tools https://developer.nvidia.com/accelerated-computing-toolkit［アクセス日 : 2022 年 11 月 7 日].

17) NVIDIA cuDNN, https://developer.nvidia.com/cudnn［アクセス日 : 2022 年 11 月 7 日].

18) UNITY EDUCATOR プラン, https://unity.com/ja/products/unity-educator［アクセス日 : 2022 年 10 月 31 日].

19) UNITY 教育機関向けライセンス, https://unity.com/ja/products/unity-education-grant-license［アクセス日 : 2022 年 10 月 31 日].

20) UNITY STUDENT プラン, https://unity.com/ja/products/unity-student［アクセス日 : 2022 年 10 月 31 日].

21) ANACONDA DISTRIBUTION, https://www.anaconda.com/products/distribution［アクセス日 : 2022 年 10 月 31 日].

22) 初めて Unity をインストールする手順について＜ Windows 編＞, 2022 年 2 月 24 日更新. https://forpro.unity3d.jp/tutorial/unity-install-windows/［アクセス日 : 2022 年 10 月 31 日].

23) コンピューターに PyTorch をインストールして構成する 2022 年 9 月 22 日更新. https://learn.microsoft.com/ja-jp/windows/ai/windows-ml/tutorials/pytorch-installation［アクセス日 : 2022 年 10 月 31 日].

24) Unity をダウンロード https://unity3d.com/jp/get-unity/download［アクセス日 : 2022 年 10 月 31 日].

25) Installing the Hub on Linux. https://docs.unity3d.com/hub/manual/InstallHub.html?_ga=2.126622349.1839134172.1669444065-522614030.1668563380#install-hub-linux［アクセス日 : 2022 年 10 月 31 日].

26)【Ubuntu Server 18.04】Anaconda をインストールする, 2019 年 5 月 4 日更新. https://paperface.hatenablog.com/entry/2019/05/04/【Ubuntu_Server_18.04】Anaconda をインストールする［アクセス日 : 2022 年 10 月 31 日].

27) Docker. https://www.docker.com［アクセス日 : 2022 年 10 月 31 日].

28) 岩田 員典, “macOS で Unity ML-Agents19 を動作させるための Docker の設定”, 2022 年 11 月 28 日 更 新. https://github.com/kazunori-iwata/ml-agents19-for-macOS［アクセス日 : 2022 年 11 月 28 日].

29) Docker で Unity ML-Agents を動作させてみた（v0.11.0 対応), 2019 年 12 月 23 日更新. https://qiita.com/kai_kou/items/6fbb8d7aa9d39820428b［アクセス日 : 2022 年 10 月 31 日].

30) Example Learning Environments, 2022 年 11 月 19 日更新. https://github.com/Unity-Technologies/ml-agents/blob/main/docs/Learning-Environment-Examples.md［アクセス日 : 2022 年 11 月 29 日].

31) Unity ML-Agents Release 18 の mlagents-learn, 2021 年 8 月 17 日更新 .https://note.

com/npaka/n/n65bc79178744［アクセス日 : 2022 年 11 月 29 日］.

32）【Unity】ML-Agents の環境をアプリ化して学習を高速化する（ML-Agents で機械学習＆強化学習をやってみる　その 6），2022 年 1 月 28 日更新．https://wakky.tech/unity-ml-agents6-app-quick/［アクセス日 : 2022 年 11 月 29 日］.

33）布留川 英一（著），佐藤 英一（編），“Unity ML-Agents 実践ゲームプログラミング v1.1 対応版（Unity ではじめる機械学習・強化学習）”，ボーンデジタル，（2020）.

34）物理ベースのキャラを強化学習向けに調整する：Unity（ML-Agents（Walker）），Unity Chan，VRM（VRoid），DAZ，2022 年 5 月 6 日更新．https://mkitys.xoxxox.net/doc/setcmm_20220504143934669886895 7.htm［アクセス日 : 2022 年 11 月 29 日］.

35）岩田 員典，“Unity ML-Agents Release 19「Walker」における報酬の影響 ”，2022 年 12 月 1 日更新．https://youtu.be/xarzJGUXpjE［アクセス日 : 2022 年 12 月 1 日］.

付録 1 Dockerfile

Dockerfile

```
FROM ubuntu:22.04
MAINTAINER Kazunori Iwata

ENV PATH /usr/bin:/usr/local/bin:$PATH

RUN apt-get -y upgrade && apt-get update && apt-get install -y curl software-properties-common vim && add-apt-repository ppa:deadsnakes/ppa

ENV LANG C.UTF-8

ENV TZ=Asia/Tokyo
RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone

RUN apt-get install -y python3.7
RUN update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.7 1
RUN apt-get -y install python3.7-distutils && apt-get install -y python3-pip git

RUN python3 -m pip install --upgrade pip
RUN pip3 install torch==1.8.1 torchvision==0.9.1 torchaudio==0.13.0 --extra-index-url https://download.pytorch.org/whl/lts/1.8/cpu

COPY setup.sh /

RUN mkdir /home/tmpuser

RUN useradd tmpuser
RUN USER=tmpuser && ¥
    GROUP=tmpuser && ¥
    curl -SsL https://github.com/boxboat/fixuid/releases/download/v0.5/fixuid-0.5-linux-amd64.tar.gz | tar -C /usr/local/bin -xzf - && ¥
    chmod 4755 /usr/local/bin/fixuid && ¥
    mkdir -p /etc/fixuid && ¥
    printf "user: $USER¥ngroup: $GROUP¥n" > /etc/fixuid/config.yml

RUN chown tmpuser:tmpuser /home/tmpuser
USER tmpuser:tmpuser
ENV PATH /home/tmpuser/.local/bin:$PATH

#WORKDIR /ml-agents/ml-agents-envs
#RUN pip3 install -e .
#WORKDIR /ml-agents/ml-agents
#RUN pip3 install -e .

# port 5004 is the port used in Editor training.
# port 6006 is the port used by tensorboard.
EXPOSE 5004 6006

ENTRYPOINT ["fixuid"]
```

付録 2 setup.sh

Docker-build.sh

```
#! /bin/sh

cd ml-agents

if [ ! -f .setup ]; then
    pip3 install -e ./ml-agents-envs
    pip3 install -e ./ml-agents
    touch .setup
fi
```

付録 4 docker-build.sh

Docker-build.sh

```
#!/bin/sh

localUID=$(id -u $USER)
localGID=$(id -g $USER)

docker run -it -u ${localUID}:${localGID} -p 5004:5004 -p 6006:6006 -v ${PWD}/ml-
agents:/ml-agents --name ml-agents ml-agents /bin/bash
```

付録 5 docker-run.sh

Docker-run.sh

```
#! /bin/sh
docker build -t ml-agents .

if [ ! -d ml-agents ]; then
    git clone https://github.com/Unity-Technologies/ml-agents.git
fi
```

付録 6 姿勢情報の取得に関する WalkerAgent.cs の修正部分

```
--- WalkerAgent.cs-org        2022-11-29 16:41:34
+++ WalkerAgent.cs            2022-11-30 12:04:26
@@ -15,6 +15,11 @@ public class WalkerAgent : Agent
     //The walking speed to try and achieve
     private float m_TargetWalkingSpeed = 10;

+    private Quaternion hipsInitialRatation; /*MOD*/
+
+    private Vector3 hipsInitialDirection; /*MOD*/
+    private Vector3 headInitialDirection; /*MOD*/
+
     public float MTargetWalkingSpeed // property
     {
         get { return m_TargetWalkingSpeed; }
@@ -85,6 +90,11 @@ public class WalkerAgent : Agent

         m_ResetParams = Academy.Instance.EnvironmentParameters;

+            hipsInitialRatation = hips.rotation; /*MOD*/
+            hipsInitialDirection = (m_OrientationCube.transform.rotation * Quaternion.Inverse(hips.rotation)) * m_OrientationCube.transform.forward;
+            headInitialDirection = (m_OrientationCube.transform.rotation * Quaternion.Inverse(head.rotation)) * m_OrientationCube.transform.forward;
+
         SetResetParameters();
     }

@@ -100,7 +110,8 @@ public class WalkerAgent : Agent
         }

         //Random start rotation to help generalize
-        hips.rotation = Quaternion.Euler(0, Random.Range(0.0f, 360.0f), 0);
+        // hips.rotation = Quaternion.Euler(0, Random.Range(0.0f, 360.0f), 0);
+            hips.rotation = Quaternion.Euler(0, Random.Range(0.0f, 360.0f), 0) * hipsInitialRatation; /*MOD*/

         UpdateOrientationObjects();

@@ -154,8 +165,10 @@ public class WalkerAgent : Agent
         sensor.AddObservation(m_OrientationCube.transform.InverseTransformDirection(velGoal));

         //rotation deltas
-        sensor.AddObservation(Quaternion.FromToRotation(hips.forward, cubeForward));
-        sensor.AddObservation(Quaternion.FromToRotation(head.forward, cubeForward));
+        //sensor.AddObservation(Quaternion.FromToRotation(hips.forward, cubeForward));
+        //sensor.AddObservation(Quaternion.FromToRotation(head.forward, cubeForward));
+            sensor.AddObservation(Quaternion.FromToRotation(hips.rotation * hipsInitialDirection, cubeForward)); /*MOD*/
+        sensor.AddObservation(Quaternion.FromToRotation(head.rotation * headInitialDirection, cubeForward)); /*MOD*/

         //Position of target position relative to cube
         sensor.AddObservation(m_OrientationCube.transform.InverseTransformPoint(target.transform.position));
@@ -252,7 +265,64 @@ public class WalkerAgent : Agent
             );
         }
```

付録 7 報酬に関する WalkerAgent.cs 変更（足を上げる）

```
        AddReward(matchSpeedReward * lookAtTargetReward);
```

を以下のように修正

```
        float footReward = 0.001F;
        float diffFootHeight = Mathf.Abs(footL.position.y - footR.position.y);
        float diffFootAndHip = 0;

        if(footL.position.y > footR.position.y){
          diffFootAndHip = hips.position.y - footL.position.y;
        } else {
          diffFootAndHip = hips.position.y - footR.position.y;
        }

        if((diffFootAndHip > 0 ) && (0.0F < diffFootHeight ) || (0.2F >= diffFootHeight )){
          footReward = diffFootHeight /0.2F * .3F;
        }

        AddReward(matchSpeedReward * lookAtTargetReward * footReward);
```

付録 8 報酬に関する WalkerAgent.cs 変更（足を上げ，頭を高く保つ）

```
        AddReward(matchSpeedReward * lookAtTargetReward);
を以下のように修正
        float footReward = 0.001F;
        float diffFootHeight = Mathf.Abs(footL.position.y - footR.position.y);
        float diffFootAndHip = 0;

        if(footL.position.y > footR.position.y){
          diffFootAndHip = hips.position.y - footL.position.y;
        } else {
          diffFootAndHip = hips.position.y - footR.position.y;
        }

        if((diffFootAndHip > 0 ) && (0.0F < diffFootHeight ) || (0.2F >= diffFootHeight )){
          footReward = diffFootHeight /0.2F * 0.3F;
        }

        float headHeight = head.position.y - hips.position.y;
        float headReward = headHeight / 0.2F * 0.3F;

        AddReward(matchSpeedReward * lookAtTargetReward * footReward * headReward);
```

PDF 敷設工事図面を用いた地理空間データの開発と解析
―中山間地域の光ファイバー網の敷設事業を事例に

蒋　湧（愛知大学地域政策学部学部）

要旨

本稿は，自治体が所持している情報通信網配線の PDF 敷設工事図面から，①配線の立地に関する地理空間情報，②配線の接続に関するトポロジー情報を抽出し，配線のコスト構造を空間的に解析できる GIS データ構造とデータ作成方法を提示し，それを用いた空間解析の結果を述べる。

近年，水道や情報通信網などを含む地域インフラの老朽化や，人口減少・少子高齢化時代における自治体インフラの維持管理への関心が高まっている。地域インフラ整備・保守計画における根拠に基づいた政策立案（Evidence Based Policy Making，EBPM）が求められるなか，それに向けた科学的な手法に注目が集められている。本稿は，愛知県設楽郡東栄町の光ファイバー網の整備事業を事例として取り上げて，敷設配線とその接続先の人口分布の関連を焦点に，情報通信システムの維持管理コスト構造における空間解析の手法を提案し，地域政策の立案に寄与することを目指す。

キーワード：① PDF・DXF・Shape file のデータ形式，② QGIS のジオレファレンス，③ GIS トポロジーのデータ構造，④公設公営の中山間地域情報通信基盤システム

1．はじめに

近年，自治体の水道配管や情報通信網などを含めた地域インフラの整備と保守計画において，GIS の活用が注目されている。その理由として，自治体インフラの老朽化や，人口減少・少子高齢化社会に伴うインフラ整備と維持における財政的な限界が挙げられる。この課題解決に向けて，自治体のインフラ整備計画における根拠に基づいた政策立案（EBPM）に関する科学的な手法が求められている。

配線の敷設工事図を用いた空間データの整備には 2 つの課題がある。一つは，PDF 工事図面から地理空間情報を抽出し，PDF ファイルをシェープファイルへ変換する課題である。もう一つは，インフラ配線間の接続情報，さらに配線の末端にある地域住民とのつなぐ情報を取り入れることである。そうすると，配線の地理空間情報に，配線間の接続情報を加えると，配線の物理的な属性と接続先の社会的な要因（人口，世帯など）を関

連付けることで，インフラ整備におけるコストの解析は可能になる。

本稿では，愛知県設楽郡東栄町の光ファイバー網の配線敷設工事を事例に，配線空間データの整備とコスト分析の手法を解説する。

愛知県北設楽郡の3町村（設楽町，東栄町，豊根村）が「公設公営」の方式で営む「北設情報ネットワーク事業」の敷設工事は2008年にスタートし，2年の工事期間を経て，2010年から運用を始めた。その背景には，政府が2000年に進めた「IT基本法」（高度情報通信ネットワーク社会形成基本法）にある。全国範囲で進められた地上デジタル波は，県境に隣接中山間地域の奥三河エリアには届いておらず，代わりに光ファイバー情報通信システムを導入した。また，人口の少ない集落が中山間地域の広域に分散していることから，光ファイバー敷設事業における民間企業の参入は極めて困難であり，結果的に「公設公営」の方式が採用された。

この光ファイバー通信システムの規模は，2008年建設当時の人口規模に基づき設計された。本稿は，2015年の調査データに基づいた分析結果であり，つまり，開業から約8年を経て，人口減少と少子高齢の進展が情報システムに与える影響を定量的に分析したい。本研究の主な結論は既に文献「1」に掲載されていたが，本稿はQGISを用いたデータ開発手法を中心に解説を行う。

2．敷設工事図面の概要

東栄町全域の光ファイバー網の敷設工事には，計295枚の工事図面（A3サイズ）がある。各々の工事図面は，ゼンリン住宅地図のメッシュ区画単位に基づき，その範囲が決められている。図1は，その295枚工事図面の中の1枚，TE33−1番の工事図面を示す。

また，図2の索引図は東栄町の全域おいて，ゼンリン住宅地図のメッシュ区画に工事図の図番が表示されている。従っ

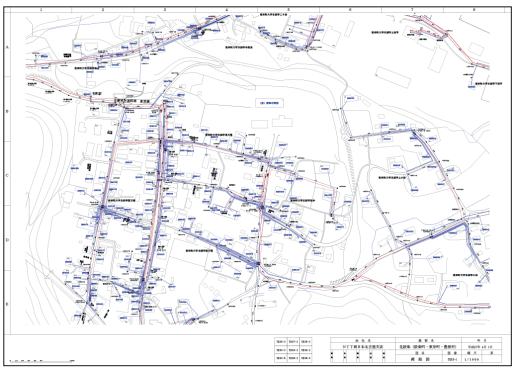


図1　TE33−1番配線の敷設工事図

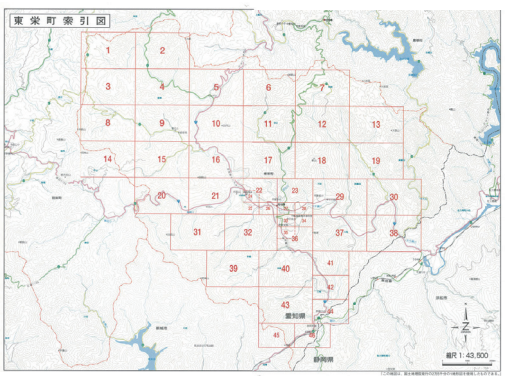


図2　全域のメッシュ区画と工程図番号の一覧

て，図２の索引図を利用すれば，各々の工事図面における地理空間上の立地が判明できる。

３．地理空間データの作成

図３には，配線の敷設工事図から地理空間データを作成する手順を示す。

敷設工事図の施工範囲は，ゼンリン住宅地図をベースにしている。まず，手順①は，ゼンリン住宅のベースマップを作成する（図４）。次の手順②は，QGISのジオレファレンスを用いて，図２の全域索引図の画像ファイルをゼンリン住宅ベースマップと重ね合わせる。その重ねることによって，ゼンリン地図のメッシュ区画と敷設工事図の図番との関連付けから，敷設工事図の地理空間情報の取得は可能になる（図６）。手順③は，TE33-1番の工事図を事例に，工事図面の画像ファイル（図１）とジオレファレ

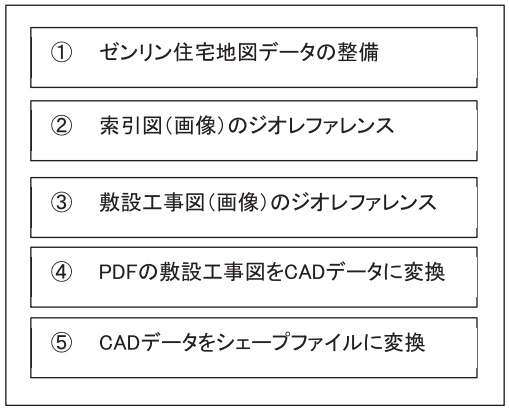


図３　配線シェープファイルの作成手順

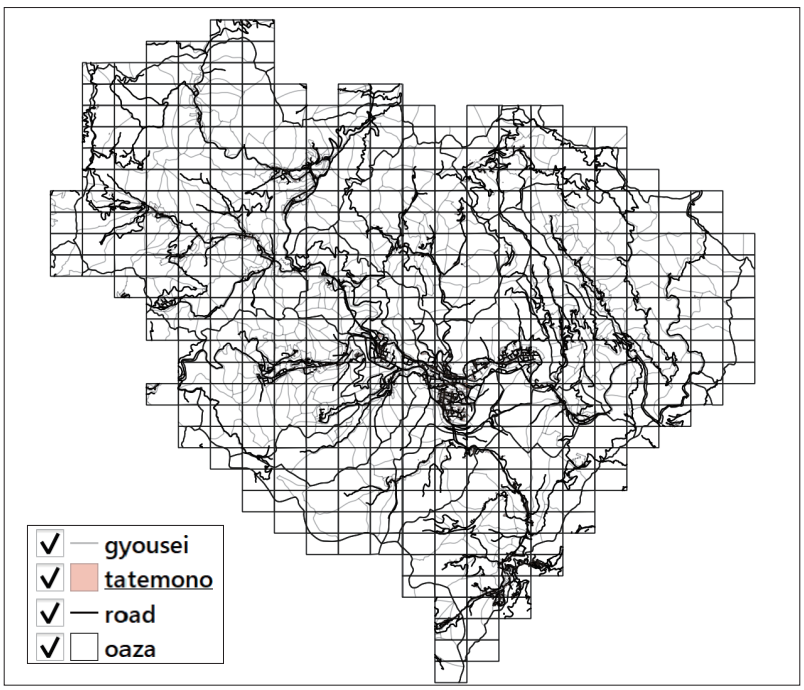


図４　ゼンリン住宅地図のデータセット

ンスを用いて，33 番のメッシュ区画と重ね合わせることで，メッシュ区画から TE33-1 の工事図面角の座標値を取得できる（図 7）。最後の手順④と手順⑤は，それぞれ PDF から CAD へ，CAD からシェープファイルへ，データ変更を行う。その際，手順③で取得した工事図面の角の座標情報が使われる。次は，上述の手順に沿って，解説を行う。

【手順 1：ゼンリン住宅地図ベースマップの作成】

ゼンリン社が提供する 2015 年の東栄町住宅データから，大字（oaza），道路（road），建物（tatemono）と行政（gyousei），計 4 つのレイヤを図 4 に示したレイヤの順に，ベースマップを作成する。ここに，愛知県のゼンリン住宅データは，日本測地系の平面直角座標系の 7 系（EPSG 30167）を使用することに注意してほしい。

【手順 2：全域配線の索引図のジオレファレンス】

PDF 形式の全域配線の索引図を入手したのであるが，QGIS ジオレファレンスを行うために，PDF ファイルを TIFF（Tag Image File Format）形式の画像ファイルに変換する必要がある。筆者は，Adobe Acrobat DC を使って，TIFF 形式の画像ファイルを取得した。

QGIS の「ジオレファレンサ」を使って，対象の TIFF 画像ファイルを選び，地理座標系は日本測地系の EPSG：30167 に設定する。次は，画像ファイルとベースマップの間に複数のグラウンドコントロールポイント（Ground Control Points，GCP）と呼ばれる照準点を入れる。ベースマップの東西南北に，4 つの方向において照準点を入れ（図 5），最後に［ジオレファレンスを開始］をクリックし，図 6 の結果が得られる。

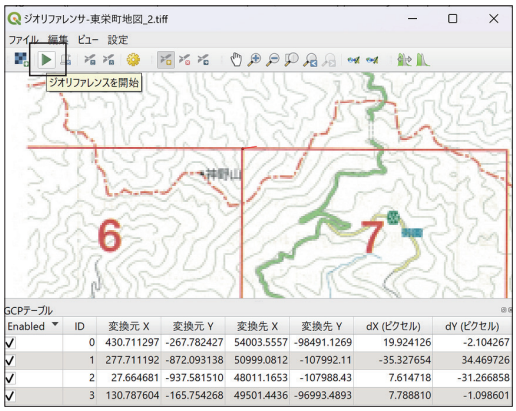


図 5　QGIS ジオレファレンスの操作

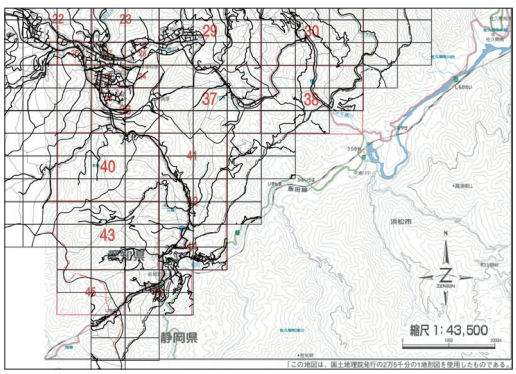


図 6　全域索引図のジオレファレンス

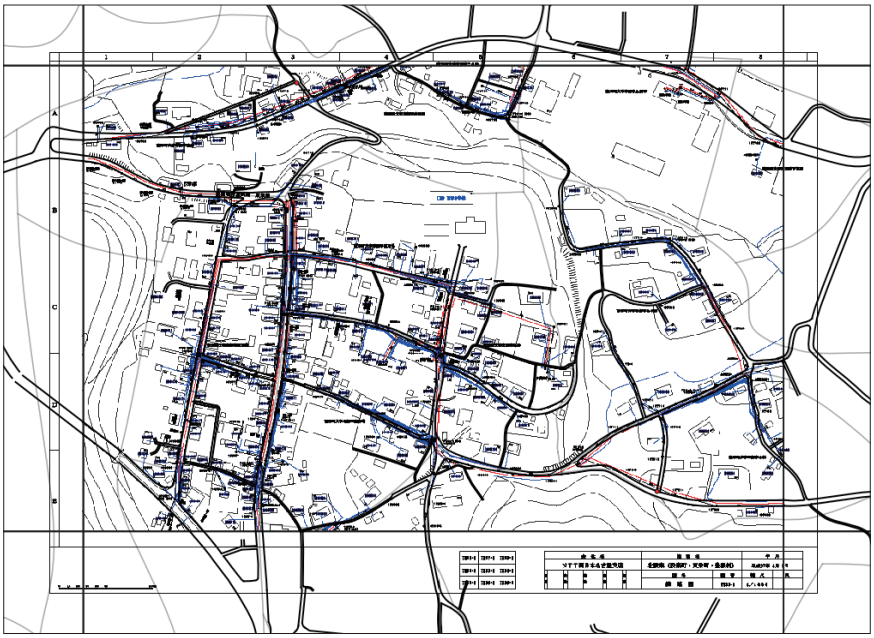


図 7　TE33–1 工程図のジオレファレンス

【手順 3：区画単位の敷設工事図のジオレファレンス】

前述のジオレファレンス手法を用いて，ベースマップ 33 番のメッシュ区画に，工事図面 TE33–1 番のジオレファレンスを行う（図 7）。そうすると，工事図面 TE33–1 の両角（左下と右下）の地理座標を取得することができる。

【手順 4：PDF を CAD データへ変換】

次は，入手した PDF の工事図面を Autodesk 社が提唱した DXF（Drawing Exchange Format）形式の CAD データへ変更する。データの変換は，Visual Integrity 社（https://visual-integrity.com/）が提供した Pdf2cad というソフトウェア（有料）を使用する（図 8）。

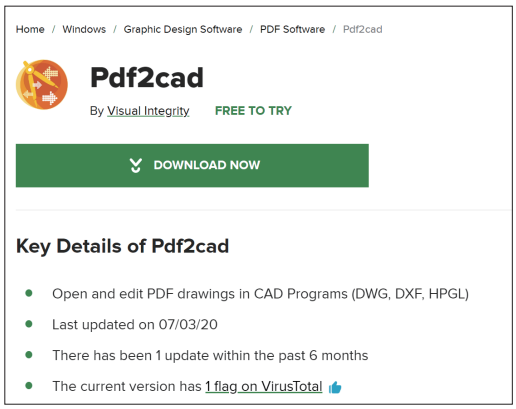


図 8　Pdf2cad の導入

データ変換の手順として，まず［Add］のボタンをクリックし，対象の PDF ファイルを選択する。次に［Options］ボタンを押し，図 9 の画面が開かれる。下部の［CAD Format］　に「DXF（Drawing Exchange Format）」のオプションにチェックを入れ

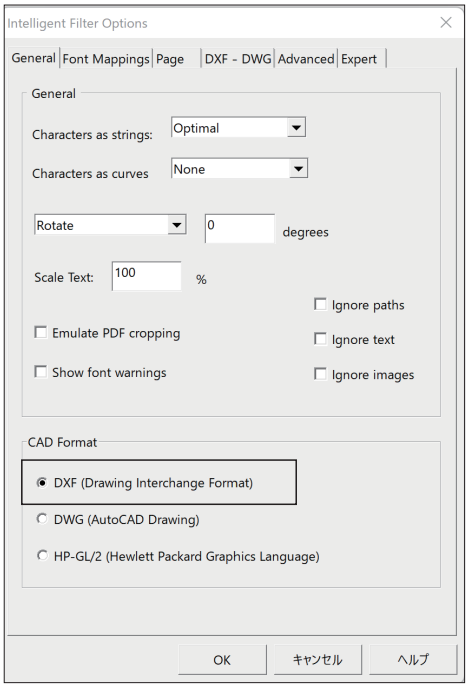


図 9　PDF を DXF の CAD フォーマットに変換

る。その後，指示に従い前に進めれば，TE33-1.pdf と同じ保存場所に，TE33-1.dxf が生成される。

【手順5:DXF をシェープファイルへ変換】

最後に，QGIS のプラグインを用いて，DXF データルをシェープファイルに変換し，図 4 のベースマップのレイヤに入れる。

まず，URL の https://plugins.qgis.org/plugins/AnotherDXF2Shape/ にアクセスし，プラグイン AnotherDXFImporter をダウンロードする（図 10）。

QGIS のメニューから，［プラグイン(P)］>［プラグインの管理とインストール］の順にクリックし，図 11 の画面が開き，［ZIP からインストール］とダウンロードしたプラグイン ZIP ファイルを選択し，［インストール］ボタンを押す。

次は，図 12 に示した方法で，手順

図10　QGISのプラグイン AnotherDXFImporter

図 11　AnotherDXFImporter をインストール

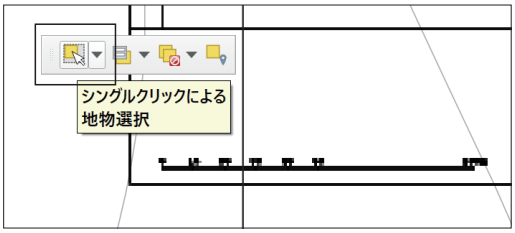


図 12　TE33-1 配線図面の角（左下）座標の取得

図 14　ジオレファレンスファイル TE33-1.WLD

③のジオレファレンス結果を用いて，TE33-1 番工事図面の角座標を取得する。選択ツール［シングルクリックによる地物選択］をクリックする（図 12）。

マウスポイント「＋」が画像ファイルの左下角に合わせ，マウス右ボタンを押す。開かれたメニューから［座標をコピー］>［Map CRS-EPSG 30167（座標値)］をクリックすると，座標値がコピーされる（図 13)。その座標値をメモ帳に貼り付け，最終的には，図 14 のように TE33-1.dxf のジオレファレンスファイルを作成し，名前 TE33-1.WLD で保存する。

DXF データのジオレファレンスファイルは，以下の形式で地理座標を記述する。

画像ファイルの座標 x，y（半角スペース）対応するベースマップ上の地理座標 x，y

TE33-1.WLD ファイルの第 1 行は，画像ファイルの左下角のジオレファレンスである。最初の（0，0）は画像ファイルの座標原点である。次の座標（48713.969，−103565.980）は図 13 で取得した画像原点に対応するベースマップの地理座標である。同様に，画像ファイルの右下角の座標を取得する。そのとき，A3 サイズの画像図面の座標は（420，0）であり，それに対応するベースマップの座標は（49536.210，−103565.952）である（図 14)。完成した TE33-1.WLD ファイルを，TE33-1.dxf ファイルと同じ場所に置く。

次は，［ベクタ］>［DXF Import /

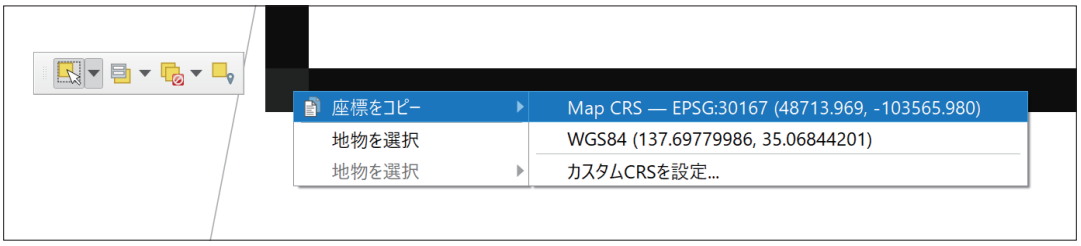


図 13　TE33-1 配線図面の角（左下）座標の取得

Convert］＞［Import or Convert］の順で Another DXF Importer を開く（図 15）。

次は図 16 に示したように設定を行う。

［Input DXF-File］＞［Browse］をクリックし，TE33-1.dxf ファイルを選ぶ

［Save as shape-files］＞チェック入れ

［Use coordinate transformation］＞チェックを入れると，図 17 の画面が現れ，TE33-1.WLD ファイルのジオレファレンスの内容を確認できる。

［use geometry collection］＞チェックを入れる

［group layer］＞チェックを入れる

［hide and use］>1.3

［Charset］>Sheft_jis

［label（Point）］＞チェックを入れる

［line］＞チェックを入れる

最後に［Import］ボタンを押す。

図 18 は，シェープファイルに変換した TE33-1 敷設工事図である。データは，ポインタ，ラインとポリゴン，3 つのレイヤに分けて，レイヤグループとしてまとめていることが確認できる。その次は，研究のニーズに合わせて，データの

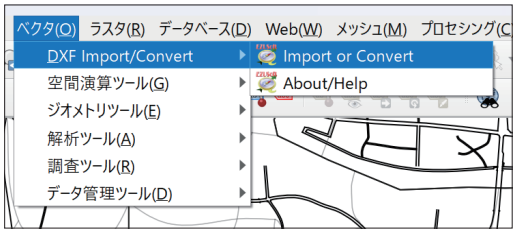


図 15　プラグイン AnotherDXFImporter を起動

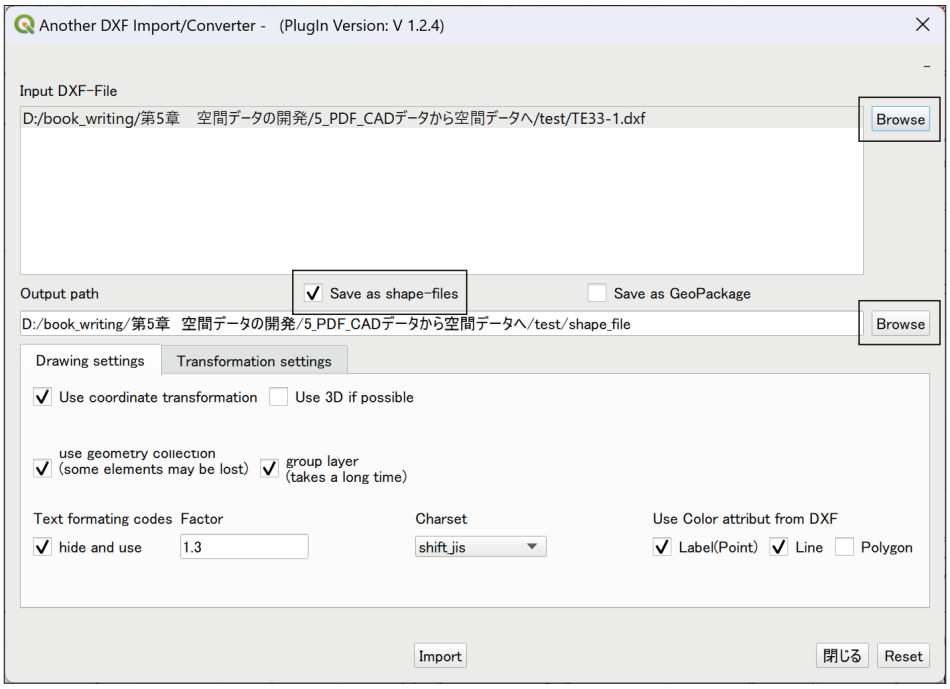


図 16　DXF からシェープファイルへの変換

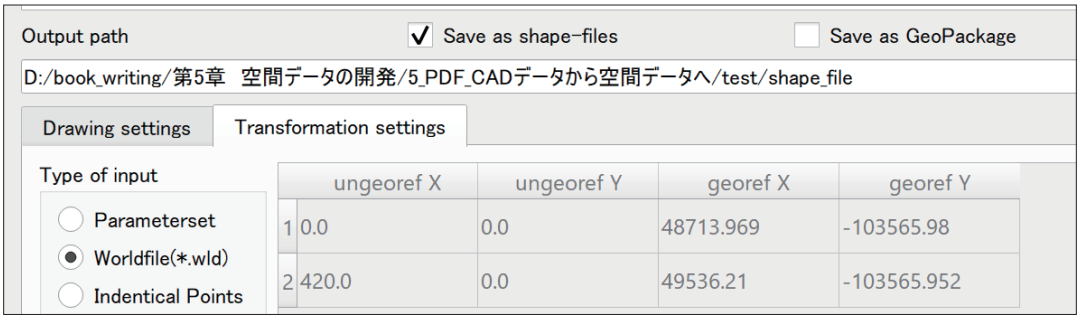


図 17　DXF からシェープファイルへの変換

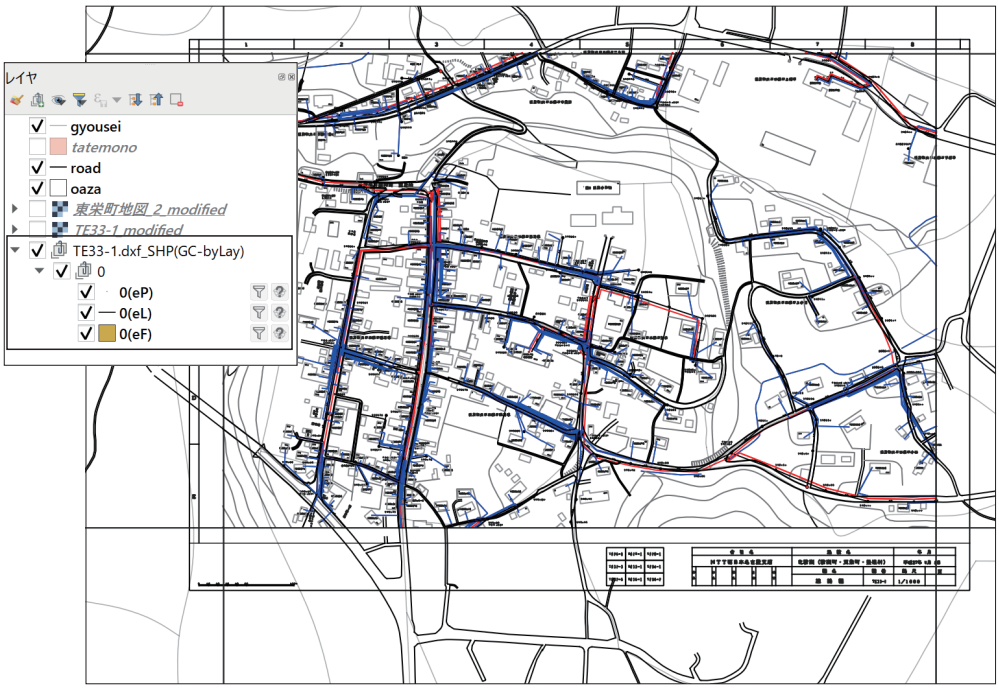


図 18　TE33-1 配線工程図のシェープファイル

取捨選択を行う。また，メッシュ区画で分断されたデータの統合と整理を含めた作業も必要になる。

4．光ファイバー網における地理空間ネットワークデータの構築

ここまで，PDF 敷設工事図から地理空間情報を抽出し，PDF ファイルをシェープファイルに変換した。次は，空間データ構造の設計を通して，配線間の接続と住民とのつなぐ情報を取り入れる。図 19 は，光ファイバーネットワークと地域住民を含めた構造図を示す。

情報通信基盤システムは，中継基地局（station），ルーター（router），幹線（trunk）

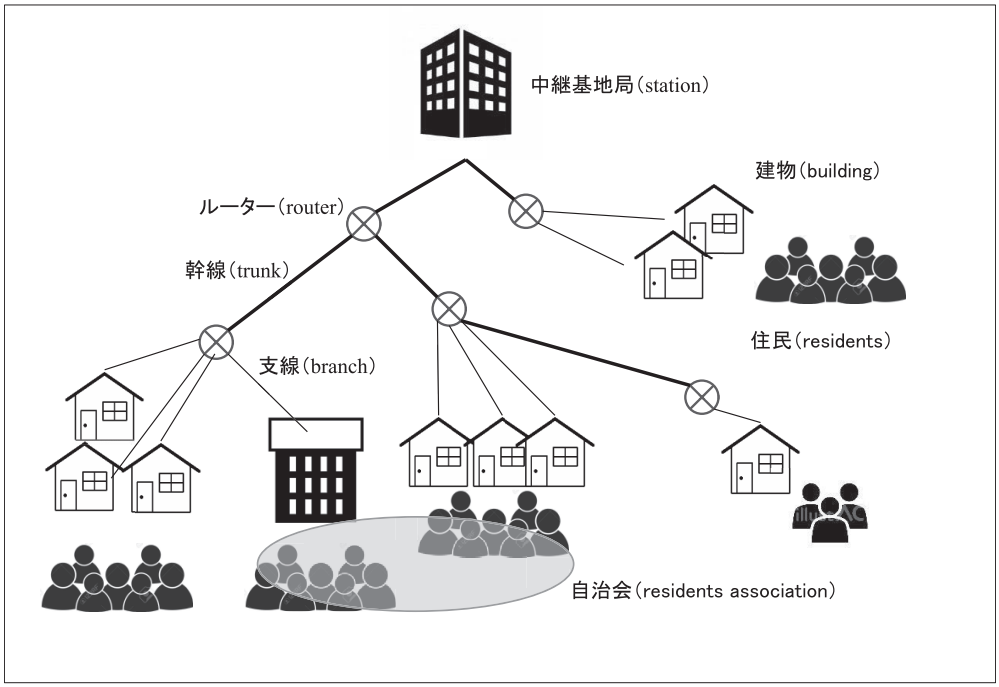


図 19　光ファイバー情報通信ネットワーク構造

と支線（branch）によって構成される。中継基地局とルーター間の配線，またはルーター間の配線は幹線と呼び，ルーターと建物間の配線は支線と呼ぶ。住民は住宅建物に住み，そこに情報通信ネットワークから支線が接続される。従って，中継基地局から情報ネットワーク末端の住宅まで，配線は「ツリー（tree）」型のネットワーク構造をもち，それを踏まえたデータ構造を設けた（図 20）。

まず，住民テール stb_residents の属性を確認してみる。ここに先頭文字 stb とは，spatial table，つまり，空間情報をもつテーブルを指す。住民の属性として，住民 ID（resident_id），世帯 ID（setai_id）と続柄（relationship）などがあり，

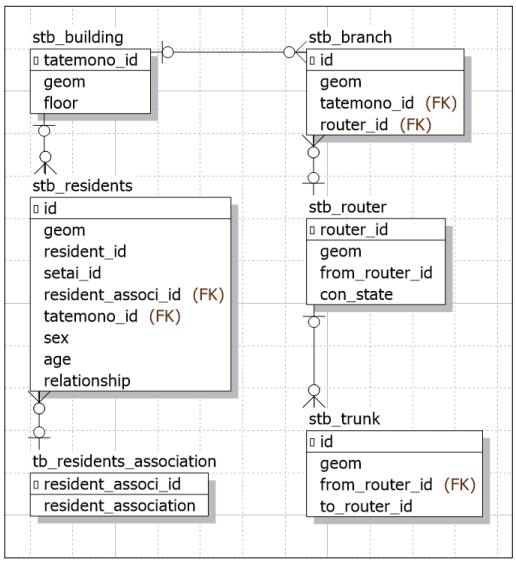


図 20　情報通信ネットワークのデータ構造

家族構成を判断できる。また，自治会 ID の外部キー（resident_associ_id）を通

して，テーブル tb_resident_association への参照ができる。ここの先頭文字 tb は，非空間情報のテーブルを指す。最後に，setai_id と建物 ID（tatemono_id）の対応で，居住者の地理空間情報は，stb_building の建物テーブルから特定できる。

次は，支線テーブル（stb_branch），ルーターテーブル（stb_router）と幹線テーブル（stb_traunk）の順に関連付けをしており，接続の情報が反映されている。特に，ツリー型のネットワーク構造の特性を生かし，常に唯一の上位ルーターを参照するように回線をつながっていく。

図 20 のデータ構造を PostgreSQL のデータベースに実装し，QGIS でマッピングした結果を図 21 に示す。

表 1 と表 2 には，それぞれ人口を含む

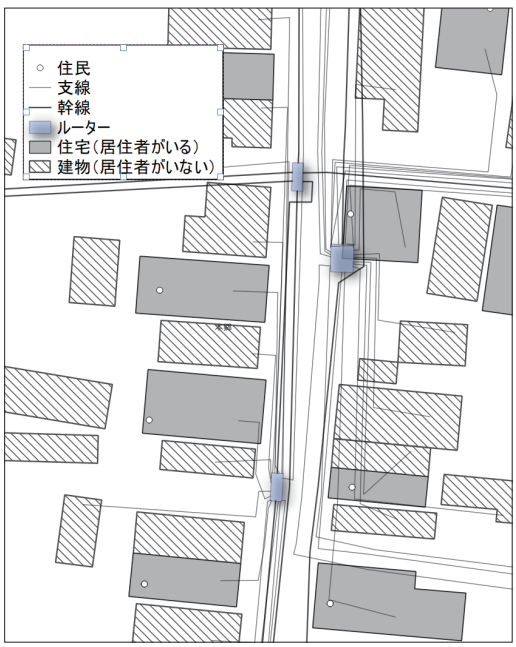


図 21　データ構造の実装結果

表 1　基礎データ

人口	3678（2015 年）
世帯	1626（2015 年）
建物	4480（2015 年）
中継基地局	1 基
ルーター	512 基
幹線	517 本，121km
支線	1903 本，133km

表 2　接続状況の集計

接続世帯数	1519（93%）
接続人口数	3414（93%）
接続建物数	1903 棟
内訳	0 世帯：853 棟（45%）
	1 世帯：880 棟（46%）
	2 世帯以上：170 棟（9%）

光ファイバー網の基礎データと接続状況の集計を示す。まず，1626 世帯をつなぐ配線の長さは 250km を超え，ネットワーク配線が広範囲に敷設されていることが窺える。次に，中山間地域の人口減少により，いわゆる「空き家」への接続数は全域の接続数の 45% に占めた。図 22 には，幹線の長さ（横軸）と接続先の世帯数（縦軸）の関係を示し，中継基地局から遠く離れる約 10％の世帯を接続するために，幹線長さの約 80% が費やされたことが判明された。

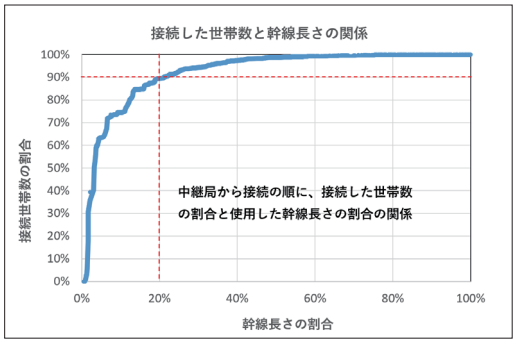


図 22　幹線の長さと接続世帯数の関係

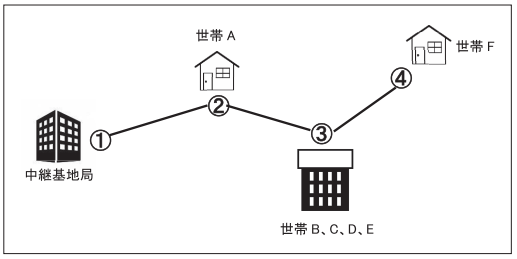


図 23　配線コスト構造の説明

5．光ファイバー網の配線におけるコスト構造と解析

図 23 には，配線コスト構造の考え方を示す。配線コストの基本構造は，配線の長さと配線を共有する世帯数のみを用いて定義する。

まず，配線③−④は世帯 F だけが使用するので，そのコストは「配線③−④の長さ／1 世帯（世帯 F)」になる。同じ考え方で，配線②−③のコストは，「配線②−③の長さ／5 世帯（世帯 B，C，D，E，F)」になる。最後の配線①−②のコストは，「配線①−②の長さ／6 世帯（世帯 A，B，C，D，E，F)」である。よって，配

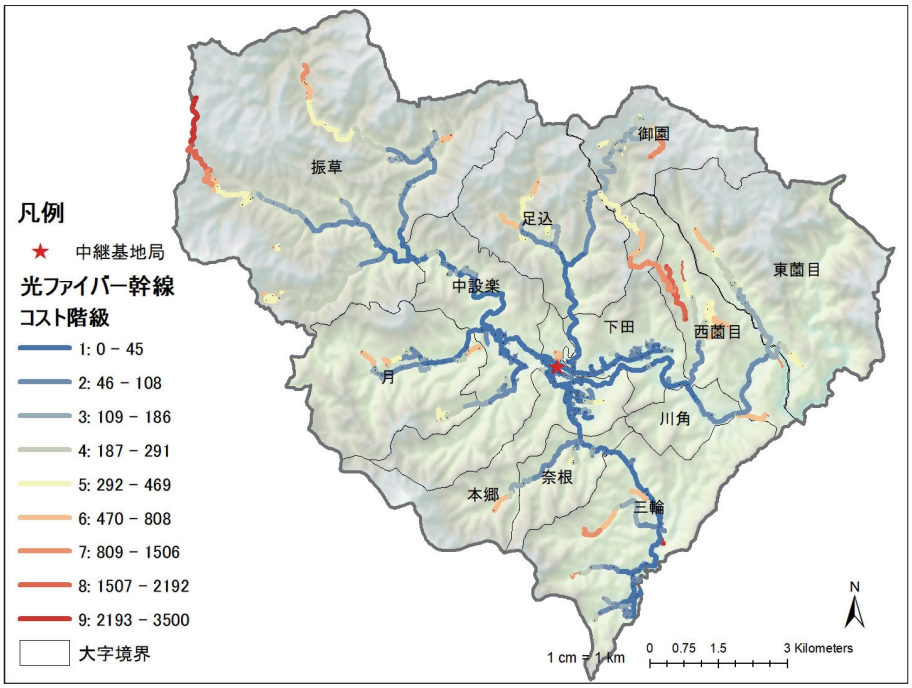


図 24　全域における敷設配線（幹線）コストの分布

線のコストは以下ように定式化できる。

$$Cost = \sum_{n} \frac{n\text{番配線の長さ}}{n\text{番配線を共有する世帯数}}$$

ここに，nは中継基地局からn番目の接続配線を指す。

図24には，算出した配線コストの高低を，青色から赤色まで，9段階のグラデーションで表現した。配線コストは，中継基地局から配線末端へ向け，配線長さの増加と共有世帯数の減少に連れ，増えていくことが確認できる。

図25は，接続世帯数（横軸）と配線コスト（縦軸）の関係を示す。まず，中継基地局から配線の末端まで，接続コストの昇順で世帯を並べ替える。次に，世帯数と配線コスト，それぞれの累積割合を計算し，その関連を図25のようにプロットした。それによると，中継基地局から最も離れている約10%の世帯の接続コストは，全域の配線コストの35%を超えた。

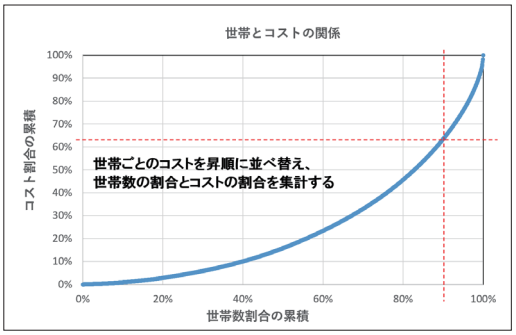


図25　配線（幹線＋支線）コストと接続世帯数の関係

図26は配線コスト（横軸）と年齢別

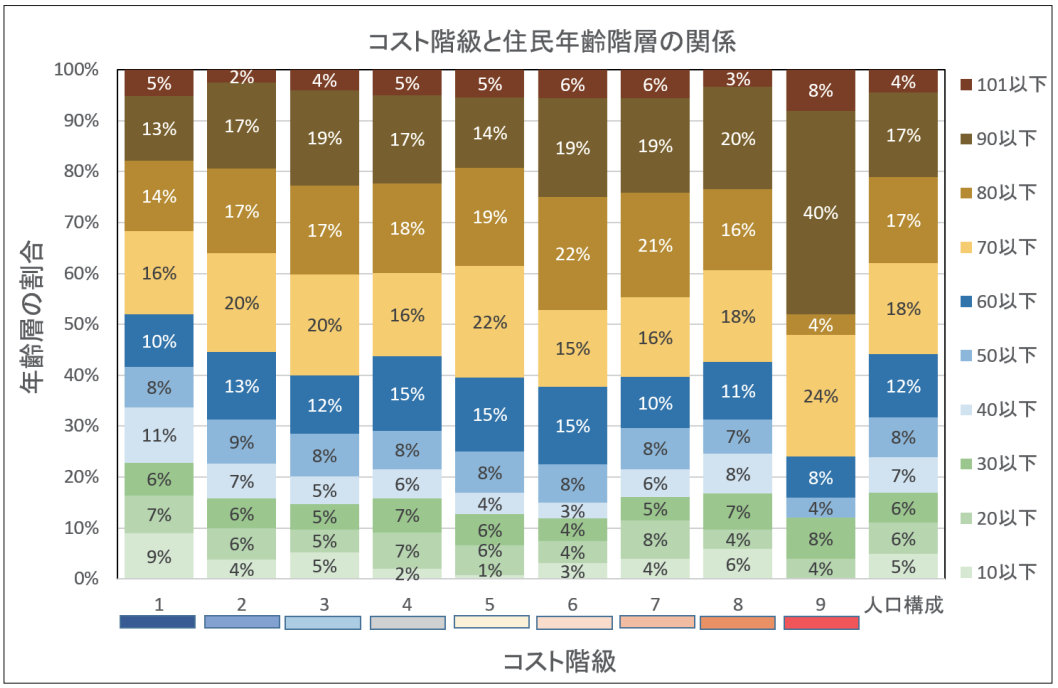


図26　配線（幹線＋支線）コストと年齢別人口分布の関係

の人口分布（縦軸）の関係を示す。図26の右側は，2015年度東栄町の10歳階級年齢別の人口分布を示す。それに対し，図の左側には，9段階のコストレベルごとのエリアにおいて，それぞれの年齢別人口の分布を示す。最も左側の低コストエリアは，中継基地局を含む町の中心部であり，町の唯一の小学校をはじめ，周辺には多くの商業施設が立地していることで，比較的に若い年齢層の人口が集中している。横軸に右方向へ推移すると，中継基地局からの距離，並びに接続コストの増加に伴い，接続先の年齢層が高くなる。コストレベル9のエリアには，60歳以上の人口の割合は76%を超えた。

6．まとめと今後の課題

本論文では光ファイバー網のPDF敷設工事図面から地理空間情報と配線接続の情報を抽出し，PDFファイルをシェープファイルへの変換方法を述べた。従来，紙ベースのマップ情報を数字化するには，QGISのジオレファレンスとデジタイジングの手法を用いて，ユーザーが手動で地物の形状を沿って点や線を引くことが必要になる。こうした作業効率の低い手法に比べ，本稿で紹介した方法は，地物データが自動的に変換されることで，作業効率が大幅に改善されることになる。一方，変換された空間データには，分析に利用できる属性は含まれていない。そのため，データの取捨選択作業は，ほぼ人力で行われ，その改善が今後の課題にしたい。

また，情報通信ネットワークのデータには，地理空間情報と配線接続情報，両方の情報をもつことで，空間トポロジーデータになる。本稿は，よく利用される道路ネットワークの空間トポロジー構造を参考し，ルーターを経由したトポロジーデータ構造を簡潔にまとめることができた。それによって，光ファイバー網の配線敷設と東栄町人口分布の特徴を踏まえた配線コストの構造が，空間的，かつ定量的に分析することができた。

参考文献

1） 蒋湧，“中山間地域の情報通信ネットワーク事業の考察”，愛知大学中部産業研究所，「地域・産業・大学研究報告」2017年報，75–89頁（2018）

付録

次の付録表は本稿に利用するデータとソフトウェアの一覧を示す。

付録表：使用するデータとソフトウェア

No	データ名称	用途	出典
1	光ファイバー網敷設工事の PDF 図面	配線システムのデータ	町役場・NTT
2	住宅地図 2015	住宅ベースマップの作成と住宅ベース人口の計算	ゼンリン株式会社
3	人口データ 2015	世帯構成の人口データ	町役場
4	その他のマップ背景データ	行政区界などの基礎データ	国土地理院
No	ソフト名称	用途	出典
1	pdf2cad	PDF ファイルから DXF ファイルへの変換	VisualIntegrity（有料）
2	Dxf2Shp	DXF ファイルからシェープファイルへの変換	QGIS の Plugin（無料）
3	QGIS 3.24	GIS ソフト	qgis.org（無料）
4	PostgreSQL 14	データベース	postgresql.org（無料）
5	PostGIS 3.2	PostgreSQL の空間拡張モジュール	postgresql.org（無料）

IoT による室内環境変化のリモートセンシング

鈴木　　臣（愛知大学地域政策学部）
深沢圭一郎（京都大学学術情報メディアセンター）
村井　孝子（純真学園大学保健医療学部）

要旨

近年の IoT 技術の発展により，計測機器の利便性が飛躍的に高まり測定データが様々な場面で利用されるようになってきた。特に，大量のデータが安定的に取得できる環境は，長期のモニタリングにおいて極めて有効である。我々は IoT を使ったセンサによって居室環境（気温，湿度，気圧，照度，騒音，TVOC 濃度，CO_2 濃度）を常時観測し，データをリアルタイムでモニタし解析するシステムを開発した。福岡県下の介護福祉団体の協力のもと，介護施設や高齢者家庭で実証実験をおこなった結果，環境の変化からは居住者の生活パタンを把握することができ，起床・就寝や外出など基本的な行動の推定も可能であった。また，2022 年 1 月 15 日に発生したトンガの海底火山噴火に伴う明瞭な気圧変動も観測された。今後は装置を多地点で展開することによって，地球物理学分野での活用や理科教育にも貢献できると考えられる。

キーワード：IoT，センサ，リモートセンシング，環境測定，Raspberry Pi

1．はじめに

「モノのインターネット」である IoT は近年急速に実用化されている。従来の情報端末だけでなく，さまざまな「モノ」がインターネットにつながることによって，業務のコスト削減と生産性の向上が期待できる。また，大量のデータが蓄積されていくことで更なる効率化が図られることになる。すでに多くの企業で AI を用いたビッグデータの解析が進められており，IoT 利用の多様化によってデータ自体の付加価値も高まっている。一般家庭においても，スマート LED やスマートロックが普及しており，IoT による「快適な生活」が実現している。AI と IoT による経済効果は 2016 年と比較して 2030 年に実質 GDP を 132 兆円押し上げると推測されており（総務省，2017），今や IoT は社会基盤を支える重要な要素といえる。

『令和 3 年度版情報通信白書』（総務省，2021）では，今後の IoT デバイス数の高成長が見込まれている分野として，医療（デジタルヘルスケア），コンシューマ，産業用途（工場，インフラ，物流），

自動車・宇宙航空が挙げられている。特に，医療現場においては，機器の動作の監視，患者のリアルタイムなデータの取得において恩恵が大きく，医療現場で使われるIoTはIoMT（Internet of “Medical” Things）とも呼ばれる。IoT（あるいはIoMT）は，医療従事者の慢性的な不足と解消につながる実用的かつ有効な手段と考えられている。

2007年に超高齢化社会に突入した日本においては，介護問題も並行して対応しなければならない。特に認知症は，介護が必要となった原因のトップであり（内閣府，2022），患者は増加の一途を辿っている。内閣府（2017）の報告では，2025年には65歳人口の5人に1人が認知症になると予測されている。また，在宅介護においても，家族の介護負担が介護離職といった深刻な社会問題に発展している。

我々は，介護者の負担軽減を目的として，非接触の環境センサを高齢者（特に認知症患者）の見守りシステムとして活用するシステムを開発した（深沢ほか，2022; 村井ほか，2021; Murai et al., 2022）。このシステムでは，環境センサが対象者の室内環境を測定し，IoT機器を通してサーバにデータを転送することで準リアルタイムな居室環境変化の把握と対象者の行動認識，室内の最高・最低気温の予測をおこなっている。

本稿では，上記の見守りIoTシステムの紹介と，得られたデータの地球物理学的計測への応用についての展望について述べる。

2．見守りシステム

図1に見守りシステムの概要を載せ

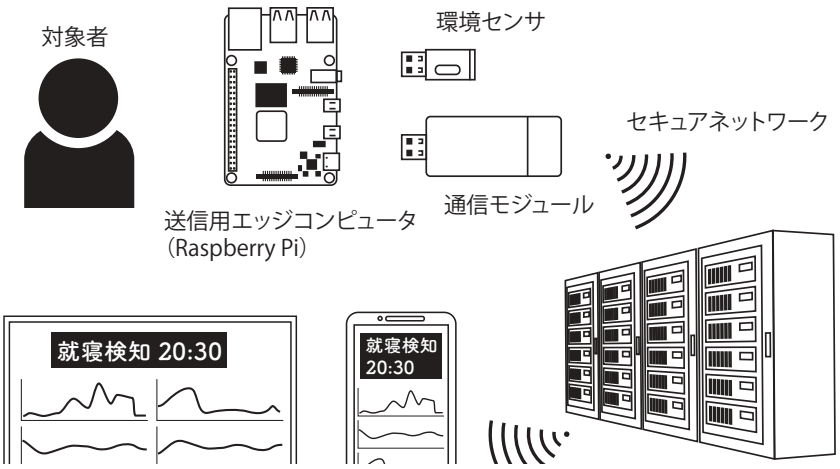


図1　見守りシステムの概要。

る。計測系は，環境センサ（オムロン 2JCIE），送信用エッジコンピュータ（Raspberry Pi 3 Model B），および通信モジュールから成る。すべて比較的安価な商用品で構成されているため入手性が良く，故障からの復帰（データの連続性）の面でメリットが大きい。環境センサは，気温，湿度，気圧，照度，騒音，TVOC 濃度，CO_2 濃度を測定できる。時間分解能は任意に設定することができ，本システムにおいては基本的に 5 分間隔でサンプリングしているが，対象によって 1 分間隔で運用している物もある。センサで得られた各種の測定データは Bluetooth LE 経由で，エッジコンピュータに送信され，暗号化された後，送信モジュールによってサーバに転送され日毎の CSV（Comma-Separated Values）ファイルとして格納される。またサーバでは，測定データを基に対象者の行動の認識が自動でおこなわれ，データの可視化と行動推定結果が指定された受信端末に表示される。

図 2 に実際の測定結果の例（2021 年 7 月 3 日：5 分分解能）を載せる。5：15 あたりにエアコン（冷房）が作動し気温が低下している様子が見られる（図 2a）。測定される湿度（図 2b）は，相対湿度であるため

（相対湿度）=（水蒸気量）/（飽和水蒸気量）

の関係式，および，Tetens（1930）の式

$$（飽和水蒸気量）= 6.1078 \times 10^{K}$$

$$K = 7.5 \times （温度[℃]）/（（温度[℃]）+ 237.3）$$

から，ある閉じた空気塊で考えると，温度が高く（低く）なると，湿度が低下（上昇）する。図 2 においても，冷房を作動させたことによる強制的な室内温度の低下に伴って湿度が 80% 近くまで上昇している。ただし 12：00 以降は気温変化を伴わない湿度増加が見られるため，気

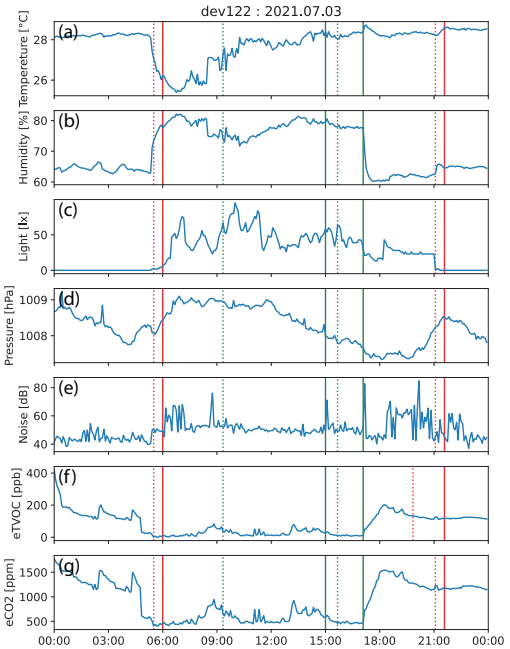


図 2　測定結果の例。2021 年 7 月 3 日における（a）気温，（b）湿度，（c）照度，（d）気圧，（e）騒音，（f）TVOC 濃度，（a）CO_2 濃度。時間分解能は 5 分。赤および緑の破線と実線で囲まれた区間はそれぞれ自動判定された起床・就寝および外出の時間を示す。

圧低下（図 2d）を考慮すると，降雨があったと推測される。このように，室内環境の変化は気象変化だけでなく対象者の行動によっても顕著に変わることがわかる（行動推定については後述する）。

なお，使用した環境センサには VOC（揮発性有機化合物）センサが搭載されているものの，簡易的な物であるためガス種までは判定できず，VOC ガスの総量換算値（eTVOC）として算出される（図 2f）。さらに二酸化炭素濃度（図 2g）は，測定された eTVOC から算出される換算値（eCO_2）である。これらのガス測定値は，人の密集，空気の澱み，換気の目安として使用できると考えられる。

装置の動作確認やイベントの発見，研究者間での情報共有のために，サーバ上のデータをウェブブラウザでプロットできる環境も整えている。Python の Web フレームワークのひとつである Bottle を使用して，描画ライブラリの Plotly でプロットを作成することで，ユーザ側が任意の装置，日付・時間のデータをインタラクティブに確認することが可能になっている。

図 3 に一例を示す。デバイス番号と日付を入力フォームに指定すると結果のプロットが表示される。プロットにマウスオーバーすることで各測定データの値が表示され，横軸を任意の時間幅に変更することも可能である。そのため，データの特徴的な変化の時間とその値をプロットから直接確認することができる。また，複数の日付のデータを重ねて表示する機能も実装している（図 4）。これは，対象者の通常の生活パタン（おおよそ，何時くらいに起床し，何時くらいに就寝する傾向にあるのか）を把握する際に有用である。なお，曜日指定（図 4 上部の DOW ウィンドウで指定）も可能になっている。この機能は，特定の曜日におこなわれる行動（通院やデイケアなどの外出）と居室環境の変化を比較する場合に用いる。

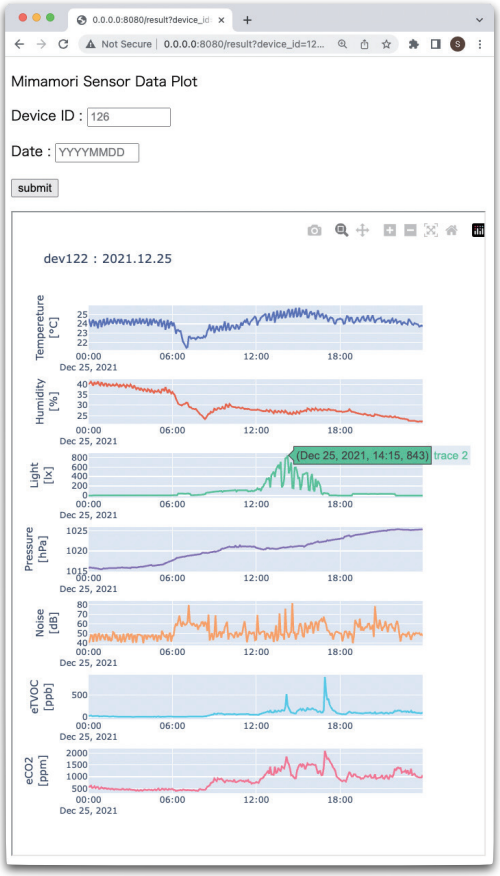


図 3　Bottle と Plotly による測定結果のウェブ表示。2021 年 12 月 25 日の例。

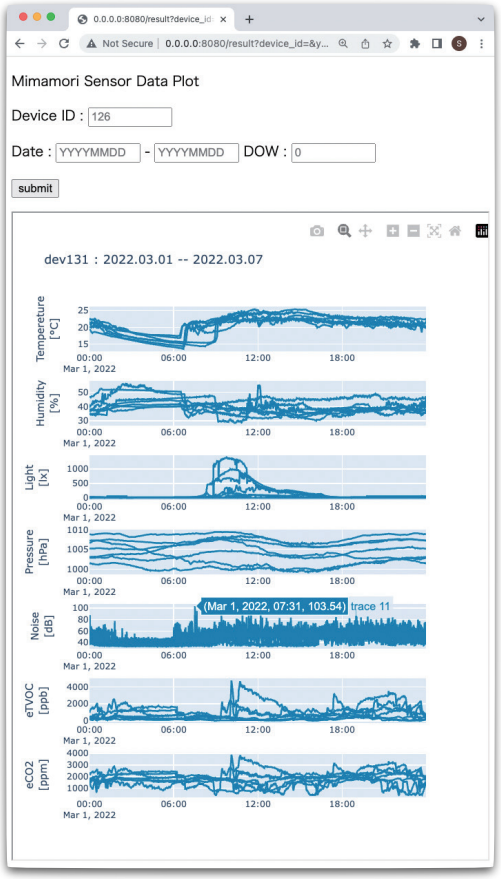


図 4　ブラウザ上で 2022 年 3 月 1 日～ 3 月 7 日（1 週間分）のマルチプロットを表示。

3. 行動推定

前節で示したように，室内環境は気象だけでなく，居住者の行動によっても大きく変化する。特に，手動でエアコンを操作する場合は，気温の変化によって居住者の安否（エアコンが操作できる状態にあること）が確認でき，さらに照度や騒音の変化からは対象者の起床，就寝，外出といった基本的な行動を推定することが可能である。

図 2 の赤線は対象者の起床と就寝（破線と実線の間で行動が起こったと判定）の時間を，緑線は対象者の外出時間（破線から実線まで）を自動で推定した結果である。部屋の照明が点けられ（照度が上がる），活動を始めた（騒音が大きくなる）タイミングで「起床」と判定し，逆に部屋の照明が消え，騒音が下がった状態が続くと「就寝」と判定される。また，部屋の騒音が小さい状態が 60 分経過した際に「外出」の判定となる。具体的な判定は，ある時刻 t において以下の条件を満たす時である。

〈起床〉

$$m_I(t-2\Delta t : t-\Delta t)+\sigma_I(t-2\Delta t : t-\Delta t) < m_I(t-2\Delta t : t)+\sigma_I(t-2\Delta t : t)$$

かつ

$$m_L(t-2\Delta t : t-\Delta t)+\sigma_L(t-2\Delta t : t-\Delta t) < m_L(t-2\Delta t : t)+\sigma_L(t-2\Delta t : t).$$

ここで，$m(a:b)$ および $\sigma(a:b)$ は，時刻 a から時刻 b までの平均値と時刻 a から時刻 b までの標準偏差を表し，添字の I と L はそれぞれ照度と騒音を示す。

〈外出〉

$$m_L(t-2\Delta t : t-\Delta t) < m_L(t-\Delta t : t)+\sigma_L(t-\Delta t : t).$$

〈就寝〉

$$m_I(t-2\Delta t : t-\Delta t)+\sigma_I(t-2\Delta t : t-\Delta t) < m_I(t-2\Delta t : t)+\sigma_I(t-2\Delta t : t)$$

かつ

$$m_L(t-2\Delta t : t-\Delta t)+\sigma_L(t-2\Delta t : t-\Delta t) < m_L(t-2\Delta t : t)+\sigma_L(t-2\Delta t : t).$$

なお，図 2 の例では，Δt=30 分として判定した。実際に装置を設置したある対象者の 20 日間の行動の自動判定の成功精度は，起床が 90%，就寝が 45%，外出が 80% であった。就寝を除き，おおむね高い精度で行動を推定できているといえる。行動推定に使用されるデータは，あくまでも居住空間の環境であり，対象者そのものデータではないため，対象者のプライバシー侵害は，カメラ映像やウェアラブル装置の解析と比較すると極めて限定的であるといえる。さらに判定精度を向上させるためには，対象者の行動や室内環境の詳細を学ぶ必要があると考えられる（深沢ほか，2022）。

また，通常の生活パタンと異なる行動（起きない，留守が続く，あるいは行動がない，など）や，特定の行動（トイレに行く，など）が検知された場合，あるいは室内の高温が予測される場合は，LINE 経由で自動的に介護者に通知を送ることで速やかに対応を促す環境も構築済みである。

4．地球科学的計測への応用

2022 年 1 月 15 日 04：10 UTC（13：10 JST）頃にトンガの海底火山（Hunga Tonga-Hunga Ha’apai）の大規模噴火が発生し，それに伴って生じた急激な気圧変化が地球規模で伝搬した。日本各地においても 1 月 15 日 20：40 JST 頃に，気象庁の地上気圧計で 2 hPa 程度の気圧変化として観測されている。

Otsuka（2022）は，日本の気象衛星「ひまわり 8 号」の水蒸気画像において噴火直後から 1 週間に渡って全球を周回するラム波が同心円状に広がっていく様子を明瞭に捉えた。ひまわり 8 号の水蒸気画像に見られたラム波が日本に到達した時刻と変動量は，地上気圧計の測定とよく一致しており，噴火に伴うラム波の伝搬が地上気圧の変化を作ったことが示された。

また，Watanabe et al.（2022）は，ウェザーニュース社が日本で展開している気象センサネットワーク（ソラテナ）の気圧計アレイデータをから，2 hPa の気圧上昇をもたらしたラム波から 2 時間遅れで到達したペケリス波による 0.1 ～ 0.2 hPa の気圧低下を捉えた。これらの噴火起源の大気波動の気圧変化が海面変動に作用したことが示唆されている。

我々の環境センサにおいても，2022 年 1 月 15 日 20：40 に明瞭な気圧上昇（変化量は約 1.5 hPa）が観測された（図 5a）。このセンサは京都大学構内（35.0°

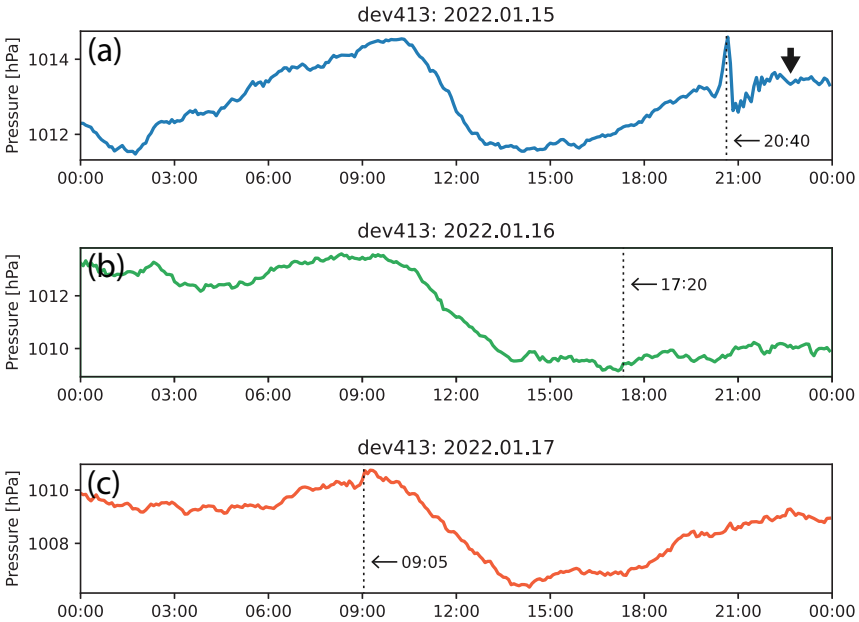


図 5　京都で観測された（a）2022 年 1 月 15 日，（b）2022 年 1 月 16 日，（c）2022 年 1 月 17 日の気圧変化。

N, 135.8° E）に設置されており，噴火した火山（20.5° S, 175.8° E）との距離は約 8018 km である。この急激な気圧変化は噴火から 7 時間 25 分後であることから，気圧変化をもたらしたラム波の伝搬速度は 300 ms^{-1} と算出できる（ただしセンサ時間分解能は 5 分である）。この速度は，Otuka（2022）のひまわり 8 号の結果（平均 310 ms^{-1}）とほぼ等しい。また，地上の音速（約 340 ms^{-1}）よりもやや遅いというラム波の特徴とも一致する。22：40（ラム波到着の 2 時間後）には，ペケリス波と考えられる 0.2 hPa の気圧低下も見られた（図 5a の太矢印で示した変化）。

さらに，火山噴火に伴って同心円状に広がるラム波が，地球の反対側から到達したと考えられる変化が 1 月 16 日 17：20（気圧上昇：約 0.2 hPa）に（図 5b），1 月 15 日に日本を通過した後に地球を一周して再来した波と考えられる変化も 1 月 17 日 09：05（気圧上昇：約 0.7 hPa）に確認することができた（図 5c）。これらの変化は 1 月 15 日に到達した第 1 波と比較すると，伝搬距離が長いため減衰しているものの，安価な気象センサでも火山噴火に伴うラム波の伝搬の様子を捉えることに成功している。

図 6 に福岡市および北九州市のセンサで観測された 2022 年 1 月 15 日の気圧変化を載せる。すべてのシステムで，20：55 前後に明瞭な 1.5–2.0 hPa 程度の気圧上昇が見られた。それぞれのセンサで観測された気圧変化の到達時間差を計算することで，日本を通過するラム波の波面を

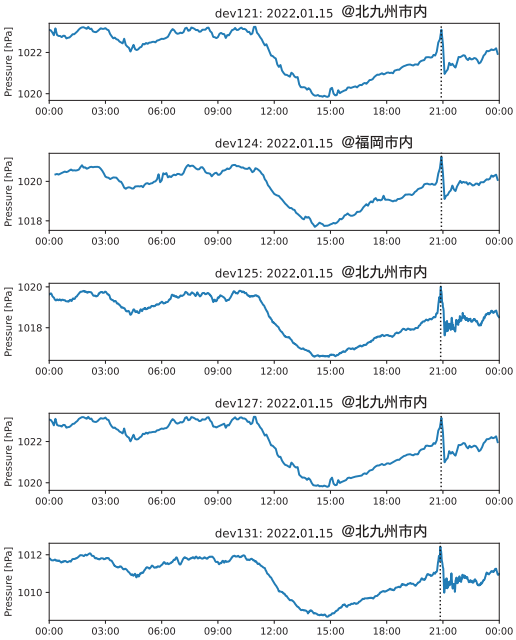


図 6　福岡市内および北九州市内に設置した 5 つのセンサで観測された 2022 年 1 月 15 日の気圧変化（点線がラム波到達時刻）。

描くことも可能である。ただし，これらのセンサは見守りシステムとして運用中であり，対象者のプライバシー保護の理由で位置情報（個人の特定につながる自宅や施設の場所）を得ることを禁じているため，水平構造を議論することはできない。

5. おわりに

我々の研究グループでは，IoT と環境センサを利用した見守りシステムを開発・運用している。環境センサによって室内環境（気温，湿度，気圧，照度，騒音，TVOC 濃度，CO_2 濃度）を測定し，データは IoT 機器を経由してサーバへ転送される。測定値は逐次的に自動で解析され，対象者の行動推定の結果を準リアルタイムに提供することに成功している。

装置は安価であり非接触かつ自律的な測定であるため，大型設備やウェアラブル装置と比較して，導入コストが抑えられ，対象者および介護者への精神的・身体的負担の軽減につながると期待される。システム自体も，堅牢性が高くメンテナンスが容易になるように設計しており，安定して長期間に渡り大量のデータを取得できるという IoT の特性が最大限に活かされているといえる。

また，環境センサでは 2022 年 1 月 15 日にはトンガの海底火山噴火に伴う気圧変動も観測された。また，地球の反対側から回り込む波動や再来波による微小変化についても認められた。到達時間や変動量は，他の計測とおおむね一致しており，地球科学計測への活用も十分に可能であることが示された。

今後は，見守り以外の用途でも設置を協力してもらい，高い時間分解能（例えば 1 分や 30 秒など）で測定するシステムを多地点で展開することでより有用な観測体制が確立できると考えられる。これらの測定は，オープンデータ化（参加者は他者のデータを自由に解析できる）することで，さらに多くのデータ集積される環境となるであろう。特に，中学・

高校に設置することが高密度ネットワーク観測網の構築には非常に有効である。台風接近に伴う気圧変化や，センサに搭載されている3軸加速度計による地震センシング，農作物の育成環境の管理など理科教育およびデータ駆動型の情報科学教育への活用も見込むことができると考えられる。

謝辞

本研究は，JSPS科研費（JP20K21739, JP18K03728）および純真学園大学共同研究費助成により実施した。データの取得は，福岡市役所保健福祉局，社会福祉法人敬愛園アットホーム福岡，社会福祉法人福岡市民生事業連盟ケアタウン茶山，株式会社ライフケアひかりの協力によるものである。ここに記して感謝の意を表す次第である。

参考文献

1）総務省:『平成29年版情報通信白書』, https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h29/index.html, 2017（参照日：2022年11月14日).
2）総務省:『令和3年版情報通信白書』, https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r03/index.html, 2021（参照日：2022年11月14日).
3）内閣府:『令和4年版高齢社会白書』https://www8.cao.go.jp/kourei/whitepaper/w-2022/html/zenbun/index.html, 2022（参照日:2022年11月15日).
4）内閣府:『平成29年版高齢社会白書』https://www8.cao.go.jp/kourei/whitepaper/w-2017/html/zenbun/index.html, 2017（参照日：2022年11月15日).
5）深沢圭一郎，鈴木臣，村井孝子：非接触センサとIoTを用いた自律的遠隔見守りシステムの研究開発，情報処理学会研究報告，2022-IS-159，Vol. 7，pp. 1–6, 2022.
6）村井孝子，深沢圭一郎，鈴木臣：環境センサを用いた要介護者療養環境および行動の認識・予測手法の検討，第9回看護理工学会学術集会，2021年10月22日–11月22日.
7）Murai T., K. Fukazawa, S. Suzuki: A Study on Predictive Detection of Excretion in Elderly Requiring Care Using Video Internet of Things, the 25th East Asia Forum of Nursing Scholars（EAFONS）Conference, 21—22 April 2022.
8）Tetens, O.: Uber einige meteorologische Begriffe., Z. Geophys, 6, 297–309, 1930
9）Otsuka, S.: Visualizing Lamb waves from a volcanic eruption using meteorological satellite Himawari-8, Geophys. Res. Lett., 10.1029/2022GL098324, 2022.
10）Watanabe, S., K. Hamilton, T. Sakazaki, and M. Nakano: First Detection of the Pekeris Internal Global Atmospheric Resonance: Evidence from the 2022 Tonga Eruption and from Global Reanalysis Data, J. Atmos. Sci., 79, 10.1175/JAS-D-22-0078.1, 2022.

日本語プログラミング言語の可読性と普及率

新村　裕太（愛知大学大学院経営学研究科）
毛利　元昭（愛知大学経営学部）
岩田　員典（愛知大学経営学部）

要旨

プログラミング言語の中でも特に有名で普及率が高いものには「Python」,「Java」,「JavaScript」,「C/C++」などがある。これらのプログラミング言語は英語がベースで，英語が母国語でない人々にとっては言語の壁がある。一方，我々が使い慣れた母国語で記述できる「日本語プログラミング言語」は，未だ認知度が低く，普及率も低い。構文が自然言語の日本語に近づけて作られた日本語プログラミング言語である「プロデル」は，プログラミング初学者に利用されやすい構文であるが，教育分野では利用されていないなど，「ミスマッチ」を起こしている。そして，そのような「ミスマッチ」を起こさず，日本語プログラミング言語の普及率を上げるには，プログラミング教育が 2020 年度から小学校，2021 年度から中学校，2022 年度から高校で必修化されたことを利用し，初等教育でプログラミング的思考が育まれた状態にし，中等教育以降で日本語プログラミング言語を活用することが重要である。

キーワード：日本語プログラミング言語，なでしこ，プロデル，ドリトル，プログラミング教育

1. はじめに

近年，様々なプログラミングツールが普及しつつあるが，我々が使い慣れた母国語で記述できる「日本語プログラミング言語」は，未だ認知度が低く，普及率も低い状況にある。日本語プログラミング言語の中で有名なものには「なでしこ」,「プロデル」,「ドリトル」というものがある。なでしこは，前身の「ひまわり」という言語の頃から事務の自動化を目標に開発されているという特徴がある。そのため，ファイル処理，画像処理，Word/Excel 連携，ネットワーク，文字列処理など，日々の定型作業を自動化するための便利な命令を 1000 以上備えている。作業の自動化をはじめ，趣味，教育，仕事など幅広く利用することが可能である。プロデルは，特徴として「読めばすぐ分かるプログラム」,「実用的な用途」,「速い」,「プラグイン」の 4 つを掲げている。また，日本語プログラミング言語の中では珍しく，教育分野での使用のためには作られておらず，汎用的なソフト

ウェア開発を目標として作られている。ドリトルは，なでしこやプロデルと比べるとより教育用途に特化しており，教育現場を中心に普及が進んでいるのが特徴である。リファレンスやマニュアルも主には授業用テキストや教師向けの授業資料という形で整備され，配布されている。

本研究では，これらの日本語プログラミング言語の特性を把握し，その普及を妨げる原因について，英語記述のプログラミング言語との可読性の比較，アンケート調査などを踏まえて考察し，普及方法や新たな活用方法を模索することを目的とする。

本稿の構成は次の通りである。まず第1章で，本稿の目的を説明する。第2章では，プログラミング言語の定義，言語処理系，プログラミング言語の種類について述べる。第3章では，日本語プログラミング言語について，なでしこ，プロデル，ドリトルの3つを用いて解説し，日本語プログラミング言語の現状を述べる。第4章では，日本語プログラミング言語の普及率が低い原因を，可読性や必要性の観点から考察し，プログラミング教育での活用についても述べる。第5章では，まとめを述べる。

2. プログラミング言語について

本章では，プログラミング言語について述べる。第2.1節でプログラミング言語について定義し，第2.2節で言語処理系，第2.3節でプログラミング言語の種類という順に述べる。

2.1. プログラミング言語

我々が普段話す日本語のような日常的に使われている言語は，自然言語と呼ばれている。プログラミング言語はそれとは別で，人の手により作られた特殊な規則に基づく言語で，形式言語と呼ばれている。言語とは，一定のルールのもとで文字，音声を組み合わせて意味を表すもの，あるいはその体系のことである。そして，プログラミングとは，コンピュータへの指示書となるプログラムを作成することである。つまり，プログラミング言語とは，コンピュータに指示をするため，文字で意味を示すことやその規則と言える。

プログラミング言語の存在意義は，人間が直接扱うには難しい機械語に代わって，より人間が扱いやすい形を提供することにある。機械語とは，コンピュータのマイクロプロセッサ（CPU/MPU）が直接解釈，実行できる命令コードの体系のことであり，0と1を並べたビット列として表され，人間が直に読み書きしやすい形式ではない。そして，コンピュータが直接理解し実行することのできる言語は，そのコンピュータの種類に固有の機械語だけである。したがって，最終的

には機械語を使ってコンピュータが行うべき作業，計算を指示しなければならない。機械語命令と一対一に対応するアセンブリ言語もプログラミング言語の一つではあるが，一般的には，C 言語や Java などより人間に解り易いプログラミング言語が使われる事が多い。アセンブリ言語は低水準言語（低級言語），C 言語や Java などは高水準言語（高級言語）と呼ばれる。高水準言語で書かれたプログラムを機械語プログラムに変換するのがコンパイラである。機械語プログラムに変換するのではなく，高水準プログラミング言語で書かれたソースコード（プログラム）を逐次解釈，実行するものをインタプリタと呼ぶ。そして，プログラムの実行に主としてインタプリタを用いるプログラミング言語をインタプリタ言語（またはスクリプト言語）と呼び，有名なものには「JavaScript」，「VBScript」，「Perl」などがある。これらの言語処理系の違いについては第 2.2 節で述べる。

AI などを用いて自然言語を機械で処理することを自然言語処理と言うが，開発には未だに難しい点も多く，中でも自然言語には言葉の「あいまいさ」，「意味の重複」が含まれていることから，感情や文脈などの解釈違いにより，勘違いをする場面が多々ある。

例として，図 1 の「いぬがねこをうんだよ」という文章を挙げる。この文章の解釈方法は，①のように「いぬ／が／ねこ／を／うんだよ」すなわち，「犬が猫を産んだよ」という区切り方と，②のように「いぬ／がね／こ／を／うんだよ」すなわち，「犬がね，子を産んだよ」という区切り方ができる。人間は犬が猫を産む訳がないと簡単に理解できるため，②が正解であると導き出せるのに対し，コンピュータの自然言語処理である予測変換などでは①が表示されてしまう。このように，「あいまいさ」や「意味の重複」があってはコンピュータに正確な処理をさせる事が難しい。そのため，プログラミング言語のような形式言語は，その様な要素はすべて取り除き，厳格な文法規則に基づいている。そのため，プログラミング言語は自然言語とは違い，コンピュータでの処理が容易となっている。

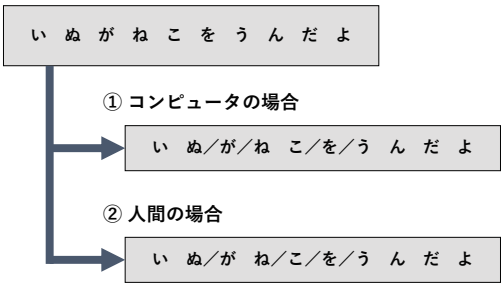


図 1　いぬがねこをうんだよ[8)]

2.2. 言語処理系

言語処理系とは，プログラミング言語で記述されたプログラムを計算機上で実行するためのソフトウェアである。その構成として，インタプリタとコンパイラ

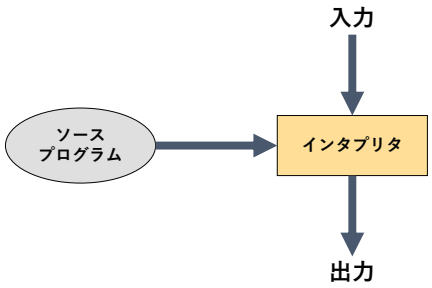


図２　インタプリタ[11)]

という２つがある。

インタプリタとは，言語を逐次意味解析しながら，その意味する動作を実行するものである（図 2)。

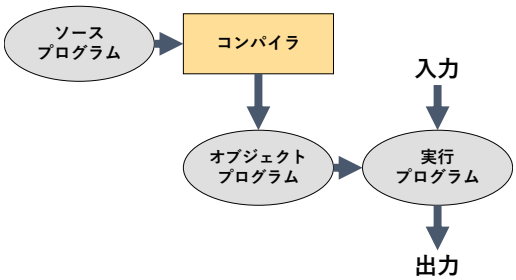


図３　コンパイラ[11)]

コンパイラとは，言語を他の言語（機械語や中間言語）に変換し，その言語のプログラムを計算機上で実行させるものである（図 3)。元のプログラムをソースプログラム，翻訳の結果と得られるプログラムをオブジェクトプログラムと呼ぶ。機械語で直接，計算機上で実行できるプログラムを実行プログラムと呼ぶ。オブジェクトプログラムがアセンブリプログラムの場合には，アセンブラにより機械語に翻訳されて，実行プログラムを得る。他の言語の場合には，オブジェクトプログラムの言語のコンパイラでコンパイルすることにより，実行プログラムが得られる。仮想マシンコードの場合には，オブジェクトコードはその仮想マシンにより，インタプリトされて実行される。

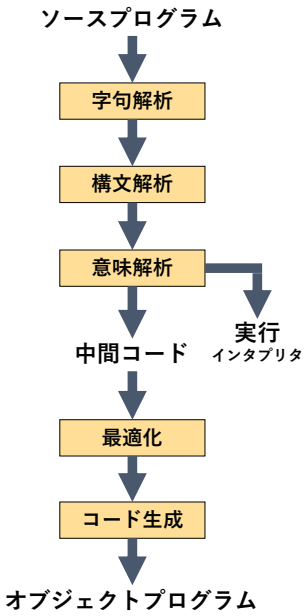


図４　言語処理系の基本構成[11)]

言語処理系の基本構成（インタプリタとコンパイラの基本構成）は主に５つに分かれる（図 4)。

1. 字句解析:文字列を言語の要素（トークン）の列に分解する。
2. 構文解析：トークン列を意味反映した構造（木構造）に変換する。
3. 意味解析：木構文の意味を解析する。インタプリタでは，ここで意味を解析し，それに対応した動作を行う。コンパイラでは，この段階で中間コード（内部的なコード）に変換する。

4. 最適化：中間コードを変形して，効率の良いプログラムに変換する。
5. コード生成：内部的なコードをオブジェクトプログラムの言語に変換し，出力する。例として，中間コードからターゲットの計算機のアセンブリ言語に変換する。

コンパイラの性能とは，如何に効率の良いオブジェクトコードを出力できるかであり，最適化でどのような変換ができるかによる。インタプリタでは，プログラムを実行するたびに字句解析，構文解析を行うため，実行速度はコンパイラの方が高速である。機械語に翻訳するコンパイラの場合は，直接機械語で実行されるために高速である。コンパイラでは中間コードでやるべき操作の全体を解析することができるため，高速化が可能である。

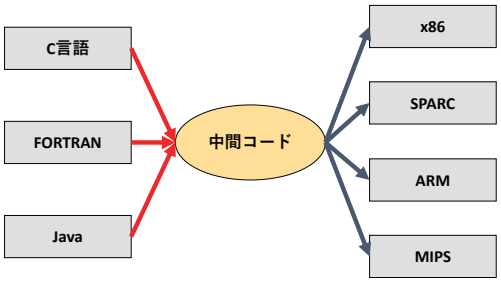


図５　中間コード[11)]

中間言語として都合の良い中間コードを用いると，様々な言語から中間言語への変換プログラムを作ることによって，それぞれの言語に対応したコンパイラを作ることができる（図５）。

2.3. プログラミング言語の種類

現在，プログラミング言語は Python や Java, C++, JavaScript などのメジャーなものからマイナーなものまで合わせると 200 種類以上存在していると言われている。その中には，企業が作成したものから個人が作成したものまで数多く存在し，文字を書くものだけではなく，視覚的に表現されたブロックを組み合わせるものや画像データを用いる言語など様々な形態が採られている。このように多種多様なプログラミング言語が存在する背景には，大きく分けて３つの要因が関わっている。

第一に，プログラミング言語ごとに得意分野が異なるという点が挙げられる。Web 開発であれば，PHP や JavaScript, Ruby，Python などのスクリプト言語がまず候補に挙がる。しかしながら，その中の１つの言語だけ覚えておけば，Web 開発以外でもすべてのシーンで使えるかというとそうではない。例えば PHP は，Web 開発に特化しており，iPhone や iPad で動作するアプリを作ることにはあまり適していない。iPhone や iPad 用のアプリを制作するには，Objective-C か Swift が主要となっている。Android 用のアプリを作るならば，Java や Java との互換性が 100% である Kotlin を用いる必要がある。また，スマートフォンアプリを開発するにも，ゲーム開発で

はC# を用いてUnity，またはC++ を用いてUnreal EngineやCocos2d-xで開発するのが現在の主流となっている。さらに，一般的なプログラミング言語とは少し異なるが，Webサイトを表示するには，HTMLという専用のマークアップ言語を使用してページ記述をし，データベースを操作する場合には，SQLというデータベースを操作する専用の言語を使用する。コンピュータの種類によっても使用する言語が異なってくる場合もある。このように，プログラミング言語ごとに専門分野や得意分野が異なり，何をどのような環境で作りたいかに応じて，プログラミング言語を変える必要がある。上記であげたプログラミング言語を含め，多くのプログラミング言語では，複数の用途に使用できるようになってはいるが，得意分野と苦手分野が明確に存在する。そのため，その時々に応じて適したプログラミング言語を使い分けることにより，効率良く開発を進める事ができるため，複数のプログラミング言語が生まれた要因となっている。

第二に，プログラミング言語の進化という点が挙げられる。ITの世界は常に進化しているイメージが強いが，それはプログラミング言語においてもそうである。プログラミング言語が使われていく中で，その言語では解決できない，もしくは解決しにくい課題が見つかっていき，それに合わせて様々な新しいプログラミング言語が開発され，その言語が主流となっていく場合もある。例えば，かつてWebサーバではPerlが標準的なプログラミング言語であったが，同じくWebサーバに特化しており，性能や書きやすさで優れたPHPが人気になり，一気にWebサーバ分野での使用率を抜かしていった。また，最近では，JavaとKotlinの間でも同様のことが起ころうとしている。KotlinがJavaと比較して優れているといわれる点は，コンパクトなコードによる安定性とセキュリティの向上，安全性を重視したコンパイラ，豊富なIDE（統合開発環境）の選択肢，生産性向上の可能性などがある。しかし，それ以上に重大な点は，KotlinとJavaの互換性が100％であることにある。つまり，JavaからKotlinに移行するのに余分な作業が必要なく，Javaのフレームワークも利用できるため，移行リスクを最小限にして，より利便性の高い言語を使用できるのである。また，2017年には，KotlinがAndroid公式言語となり，2019年には，Android上でKotlinファーストの推進を発表し，Java以上にAndroid上で優先されるプログラミング言語となっている。このような事が要因となり，Kotlinは近年急速にユーザーを伸ばし，Javaに取って代わろうとしている。このように新しいプログラミング言語を作ることによって，これまで時間のかかっていた処理を簡単に実現でき

るようになるなど，効率性，利便性の向上を見込めることもあるため，プログラミング言語の種類が増加していく要因となっている。

第三に，プログラミング言語の多様性という点が挙げられる。プログラミング言語は誰でも自由に作れ，その新しく作られる言語は，第二でも述べたように何かしらの課題解決のために生まれている事が多い。そのため，同様の用途のプログラミング言語同士であっても，完全に近い進化であるかどうかに関わらず，劣っている点もあれば，優れている点もあるといった状態で並立して生き残っている。それぞれ独自の事情やセールスポイントがあるからこそ，並立していても何かしらの使い分けが存在し，共存が可能になっているという事である。さらに，IT の分野では一般的に独占が起きづらいと言われており，常に多様な選択肢が存在する。例えば，スマートフォン一つとっても，Apple の iOS や，Google の Android など複数の OS が存在し，どの OS にも良い点があり，一概にどちらが優れているとは言えない。そして，それぞれの OS に合わせた言語も発表され，プログラミング言語の多様性も生まれているのである。その他にも，用途的に並立した言語同士であっても個人の書き方やプラットフォームなどに関する好みによって分かれてくる。例えば，Java と C# はどちらもデスクトップアプリを開発したり，Web アプリを開発したりすることが可能で，静的型付けを行うオブジェクト指向のプログラミング言語という点や，一度は中間形式に変換される点など，細かい点でも似ている点がある。しかし，その他の書き方やプラットフォームなどでは異なっている点も多く，Java と C# のそれぞれに数多くの支持者がおり，現在も開発継続されている。このように，用途的に並立しても消えずに存在することが多いという，プログラミング言語の多様性も，複数のプログラミング言語が存在する要因となっている。

上記で挙げたプログラミング言語以外のものとして，本研究の題材でもある，「日本語プログラミング言語」という種類がある。第 1 章でも述べたように，日本語プログラミング言語は，未だ認知度が低く，普及率も低い状況にある。日本語プログラミング言語の中で有名なものには「なでしこ」，「プロデル」，「ドリトル」というものがある。この 3 つのような日本語プログラミング言語同士でも，様々な用途や特徴があるが，その中でもほとんどの日本語プログラミング言語に共通していて，最も使用率の高い用途が，プログラミング初学者や学生向けのプログラミング教育である。日本人にとって日本語で読み書きできるという点は最大の利点であるため，プログラミング入門用の言語として期待できる。ま

た，日本語プログラミング言語の先駆けとなったMindや，熱心に開発が続くなでしこ，プロデル，ドリトルはいずれもトイ（知育玩具用）言語ではなく，十分に本格的な言語であり，教育分野以外でも多くの業務に使用可能となっている。

3. 日本語プログラミング言語について

本章では，第3.1節で日本語プログラミング言語について，第3.1.1項でなでしこ，第3.1.2項でプロデル，第3.1.3項でドリトル，第3.2節で日本語プログラミング言語の現状という順に述べる。

3.1. 日本語プログラミング言語

日本語プログラミング言語とは，日本語風のプログラミング言語のことである。ひらがなやカタカナ，漢字などを用いて日本語でソースコードを書き，アプリケーション開発を行うことができる。

日本語プログラミング言語の歴史は意外と古く，1982年に発表されたぴゅう太には，日本語BASIC（G-BASIC）が搭載されていた。1983年には，ワープロ感覚で日本語をもちいてプログラミングができればという考えで開発された和漢が開発された。1985年には，Forthの語順が日本語に近いことに注目したMindが開発された。そして，2000年には，TTSneo，プロデル，ひまわり，ドリトルなど，インタプリタ型の日本語プログラミング言語が次々とフリーソフトとして発表された。

日本語プログラミング言語の最大の利点は，我々が使い慣れた母国語である日本語で記述できるという点である。しかしながら，日本語プログラミング言語は，未だ認知度が低く，普及率も低い状況にある。そのため，「日本語プログラミング言語の可読性と普及率」について考察する必要がある。まず初めに，日本語プログラミング言語の中でも有名である，「なでしこ」，「プロデル」，「ドリトル」について特徴や使用用途を述べる。

3.1.1. なでしこ

ホームページでの紹介には，「『なでしこ3』は，日本語をベースに開発されたプログラミング言語です。主にWebブラウザで動かすことを目的に開発されています。ブログやWebサイトに組み込んで使えます。また，OSや環境を問わず，PC/スマホ/タブレットとさまざまな環境で動きます。作業の自動化をはじめ，趣味，教育，仕事などに利用できます。」とある[14)]。

このようになでしこは，日本語で記述をするため，日本人のプログラミング初学者や，学生，英語の苦手な人などに向けて作られている。しかし，元々，前身のひまわりという言語の頃から事務の自動化を目標に開発されている。そのた

め，ファイル処理，画像処理，ネットワーク，Word/Excel 連携，文字列処理など，日々の定型作業を自動化するための便利な命令を 1000 以上備えている。そして，なでしこは 2004 年に公開されて以降，2017 年には Web ブラウザでも動かせる v3（CoffeeScript や TypeScript のように，JavaScript に変換されて実行される広義の altJS）が公開されるなど，その後も活発に開発が続けられている。

3.1.2. プロデル

ホームページでの紹介には，「プロデルは，日本語でプログラムが書けるプログラミング言語です。次のような特徴を持っています。プロデルの文法は日本語文のように書けるように設計されています。プログラム例文を見ればすぐに何をしているか理解できます。バッチ処理や画像描画，GUI 部品の機能が用意されています。Excel，CSV 形式操作，ODBC 接続もできます。ちょっとしたツールやゲーム作りも可能です。TTSneo よりも高速に動くようになりました。大量のデータ処理も短時間で実行できます。また MSIL コンパイラも搭載していますので，コンパイルで更なる高速化も可能です。プラグインで FeliCa や Twitter との連携機能を利用できます。また C# を使って独自のプラグインを作ることもできます。」とある[15]。

このようにプロデルは，「読めばすぐ分かるプログラム」，「実用的な用途」，「速い」，「プラグイン」の 4 つを特徴として挙げられる。そのため，日本語プログラミング言語の中では珍しく，教育分野での使用のためには作られておらず，汎用的なソフトウェア開発を目標としている。また，プロデルにはプロデル Web サーバという，プロデルで開発された Web アプリケーションを動作させるための専用 Web サーバが用意されている。この Web サーバを利用することにより，プロデルで Web アプリケーション開発の実地やサービスの公開が可能となる。プロデルは同じく日本語プログラミング言語の TTSneo を前身とし，仕様を一新，アップグレードする形で公開された言語である。TTSneo の公開は 2002 年，さらに，TTSneo の前身である TTS が公開されたのは 2000 年であるため，TTS の公開から数えると，プロデルは 20 年以上もの歴史を持つ。

3.1.3. ドリトル

ホームページでの紹介には，「ドリトルは教育用に設計されたプログラミング言語です。中学校・高校の教科書や副教材などに採用されています。小学校（総合・算数・理科・音楽など），中学校（技術科の計測制御・双方向コンテンツ），高等学校（情報の科学・社会と情報，情報 I・情報 II），大学（プログラミング入門，IoT）などでご利用ください。」とある[16]。

このようにドリトルは，なでしこやプロデルと比べ，より教育用途に特化し，教育現場を中心に普及が進んでいるのが特徴である。リファレンスやマニュアルも主には授業用テキストや教師向けの授業資料という形で整備され，配布されている。

3.2. 日本語プログラミング言語の現状

日本語プログラミング言語の現状について，「日本語で記述できるプログラミング言語の存在を知っていますか？」という内容の認知度アンケート調査を行った（図 6）。アンケート調査を行った対象は，いずれかのプログラミング言語を授業もしくは，職業関連などで一度は触れたことがある方のみ，100 名である。アンケート対象の年齢には特に制限は設けず，年代ごとに認知度が大きく変わるという事もなかったため，年齢は表記しない。

アンケート結果は，「はい」と回答した方が 12 名，「いいえ」と回答した方が 88 名となった。ただし，アンケート対象 100 名の中では，いずれかのプログラミング言語に職業関連などで触れたことがある方が 78 名，授業で触れたことがあるだけの方が 22 名。職業関連の方が比べて多いため，「はい」が多少多い可能性も考えられる。いずれかのプログラミング言語に一度は触れたことがある方のみに絞っても認知度がかなり低く，どのプログラミング言語にも一度も触れたことがない方を含めれば，更に認知度が低くなる結果が出ると考える。

このように，日本語プログラミング言語は我々の母国語である日本語で記述できるプログラミング言語であるにも関わらず，未だ認知度が低く，普及率も低い状況にある。そのため，「日本語プログラミング言語の可読性と普及率」につい

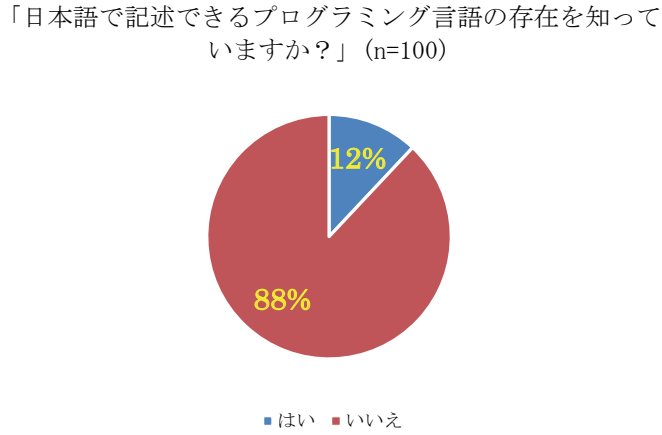


図 6　プログラミング言語の認知度調査

て考察する必要がある。そして，普及率が低い原因について考察するにあたり，日本語プログラミング言語と英語記述のプログラミング言語との可読性の比較や，日本語プログラミング言語同士での可読性の比較を行う必要がある。第 4 章では，これらの調査を行った上で，普及率の高い言語との活用目的，環境，構文などの違いを見つけ出し，普及率にどの様な影響を及ぼしているか考察し，日本語プログラミング言語の普及率を上げる方法について述べる。

4. 日本語プログラミング言語の可読性と普及率

本章では，第 4.1 節で英語記述のプログラミング言語との可読性の比較，第 4.2 節で日本語プログラミング言語同士での可読性の比較，第 4.3 節で日本語プログラミング言語の必要性，第 4.4 節で普及率が低い原因，第 4.5 節でプログラミング教育での活用，第 4.6 節で初等教育からのプログラミング教育の必要性，第 4.7 節でプログラミング教育の実態という順に述べる。

4.1. 英語記述のプログラミング言語との可読性の比較

日本語プログラミング言語と英語記述のプログラミング言語との可読性の比較をするにあたって，① Java と②なでしこについてアンケート調査を行った。両者ともアンケート内容は「FizzBuzz 問題」について記述された簡易的なソースコードである。アンケート調査を行った対象は，プログラミング言語に触れた経験がない方も含めた 100 名である。アンケート対象の年齢には特に制限は設けず，年

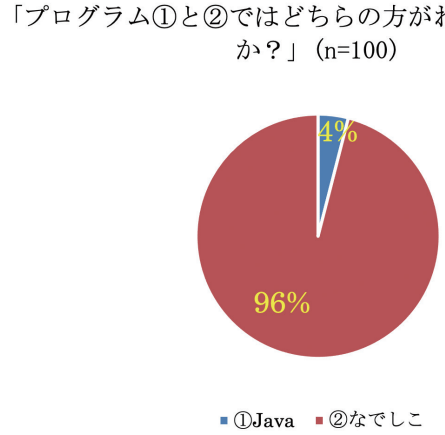


図 7　英語記述のプログラミング言語との可読性の比較

代ごとに認知度が大きく変わるという事もなかったため，年齢は表記しない。以上を前提として，「プログラム①と②ではどちらの方がわかりやすいですか？」という内容でアンケート調査を行った(図7)。また，そのように回答した理由についても同時に集計を行った。

アンケート結果は，「①」と回答した方が4名，「②」と回答した方が96名となった。ほぼ全員がJavaよりなでしこの方がわかりやすい，すなわち英語記述のプログラミング言語より日本語プログラミング言語の方が，可読性が高いといえる結果となった。「①」と答えた理由としては，「Javaのような英語記述のプログラミング言語は使い慣れているため」，「普段からJavaを使うことが多いため」，「日本語プログラミング言語は初めて見たので少し違和感があるため」，「知らないプログラミング言語よりも知っているプログラミング言語の方がわかりやすいため」という4つが挙がった。この事から，「①」と回答した4名ともに共通する要因は，Javaとなでしこの認知度や普及率の差であると考える。日本語プログラミングは英語記述のプログラミング言語と比べ，遥かに認知度や普及率で劣っているため，当然といえる意見である。そのため，これらの意見は，日本語プログラミング言語の普及率がより高くなった際には解決へ向かうと考える。

4.2. 日本語プログラミング言語同士での可読性の比較

日本語プログラミング言語の中でも有名な「なでしこ」，「プロデル」，「ドリトル」の3つの中では，なでしことプロデルは自然言語の日本語に近く，ドリトルはより英語記述のプログラミング言語に近い性質を持っている。第4.1節，日本語プログラミング言語と英語記述のプログラミングとの可読性の比較では，Javaとなでしこを用いてアンケート調査を行ったが，母国語である日本語で記述されているからという理由で，日本語プログラミング言語の方が英語記述のプログラミング言語よりわかりやすいと回答した方がほとんどであった。しかしながら，母国語である日本語プログラミング言語の中でも，構文が自然言語の日本語に近づけて作られた言語と，英語記述のプログラミング言語に近づけて作られた言語で分かれている。

そこで本節では，日本語プログラミング言語同士での可読性の比較をするため，①なでしこと②ドリトルについてアンケート調査を行った。アンケート内容は「FizzBuzz問題」について記述された簡易的なソースコードである。アンケート調査を行った対象は，プログラミング言語に触れた経験がない方も含めた100名である。アンケート対象の年齢には特に制限は設けず，年代ごとに認知

度が大きく変わるという事もなかったため，年齢は表記しない。以上を前提として，「プログラム①と②ではどちらの方がわかりやすいですか？」という内容でアンケート調査を行った（図 8）。

アンケート結果は，「①」と回答した方が 45 名，「②」と回答した方が 55 名となった。一見，ほぼ同じようにも見えるが，「①」と回答した方は，ほとんどがプログラミング言語に触れた経験がない，もしくはプログラミング言語を授業で触れたことがある程度であった。それに対し，「②」と回答した方は，プログラミング言語に職業関連などで触れたことがある，かなり英語記述のプログラミング言語を使い慣れている方がほとんどであった。すなわち，なでしこのように構文が自然言語の日本語に近づけて作られている言語は，プログラミング初学者や，プログラミング言語の学習途中の学生などに受け入れられやすい傾向があると考える。そして，ドリトルのように，構文が英語記述のプログラミング言語に近づけて作られている言語は，いずれかのプログラミング言語を習得している，もしくはプログラミング言語の学習過程にあるが，既にある程度十分な知識を持っている方に受け入れられやすい傾向があると考える。そのため，教育分野での普及では，構文を自然言語の日本語に近づけて作られた，学習用の日本語プログラミング言語がカギとなり，仕事で実際に使用するなどの本格的な業務分野での普及では，構文を英語記述のプログラミング言語に近づけて作られた，ソフトウェア開発環境の整った日本語プログラミング言語がカギとなる。

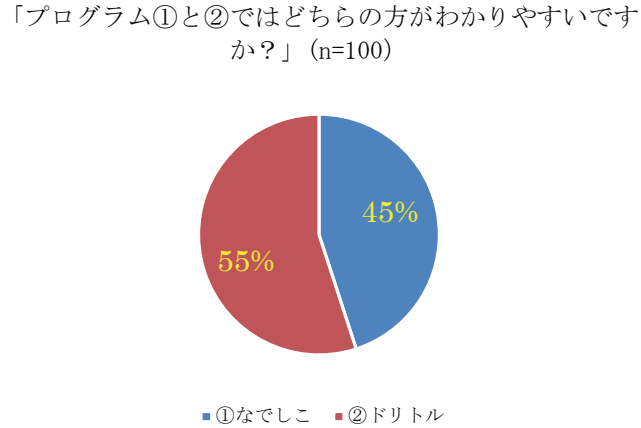


図 8　日本語プログラミング言語同士での可読性の比較

4.3. 日本語プログラミング言語の必要性

日本語プログラミング言語の必要性として，日本人のプログラミング初学者や，学生，英語の苦手な人などにとって学習コストが低いという事ももちろん挙げられるが，必要になってくる要因の中でも最も大きなものとして，言語間距離がある。

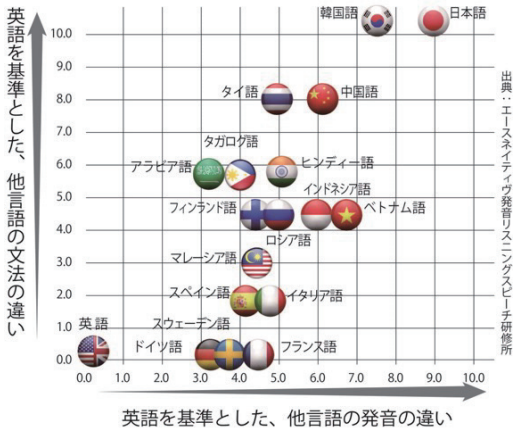


図 9　言語間距離[18)]

インド・ヨーロッパ語族の言語として代表的なものには英語がある。しかしながら，図 9 からわかるように，日本語は英語からかけ離れた言語である。そのため，プログラミング言語においては，他の国に比べても，英語記述のプログラミング言語ではなく，母国語である日本語の言語の必要性が高くなってくる。また，言語間距離の原因として，日本語との単語や文法，発音，文化，風習などの違いが考えられる。その中でも，プログラミング言語について考察するため，文法に着目すると，日本語は SOV 型であるが，インド･ヨーロッパ語族の言語（ドイツ語，オランダ語を除く）は SVO 型である。文法の違いは可読性の高さについて考えていく際にも重要な要素になり得る。

4.4. 普及率が低い原因

ここまでに述べたことを踏まえた上で，日本語プログラミング言語の普及率が低い原因として，日本語プログラミング言語を取り巻く環境で起きている「ミスマッチ」が挙げられる。第 2.3 節，プログラミング言語の種類でも述べたように，プログラミング言語ごとに専門分野や得意分野が異なる。さらに，コンピュータの種類によっても使用する言語が異なってくる場合もあるため，何をどのような環境で開発するかに応じて，プログラミング言語を変える必要がある。また，第 4.2 節，日本語プログラミング言語同士での可読性の比較で挙げたアンケート結果より，教育分野での普及では，構文を自然言語の日本語に近づけて作られた，学習用の日本語プログラミング言語がカギとなり，仕事で実際に使用するなど本格的な業務分野での普及では，構文を英語記述のプログラミング言語に近づけて作られた，ソフトウェア開発環境の整った日本語プログラミング言語がカギとなる。

まず初めに，教育分野での普及について述べる。日本語プログラミング言語の中では，ドリトルが，なでしこやプロデルと比べて最も教育分野に特化しており，教育現場を中心に普及が進んでいるのが特徴である。また，リファレンスやマニュアルも主には授業用テキストや教師向けの授業資料という形で整備され，配布されている。このような事から，ドリトルを教育用途で利用するのに適した環境は整っているように見える。しかしながら，ドリトルは構文を英語記述のプログラミング言語に近づけて作られた言語であり，それを利用すると考えられる層は，アンケート結果より，いずれかのプログラミング言語を習得している，もしくはプログラミング言語の学習過程にあるが，すでにある程度十分な知識を持っている層である。

次に，業務分野での普及について述べる。日本語プログラミング言語の中では，プロデルが，なでしこやドリトルと比べて最も業務分野に特化している。プロデルは，日本語プログラミング言語の中では珍しく，教育分野での使用のためには作られておらず，汎用的なソフトウェア開発を目標としているのが特徴である。バッチ処理や画像描画，GUI 部品の機能が用意されている，Excel，CSV 形式操作，ODBC 接続が可能，ツールやゲーム作りが可能，大量のデータ処理を短時間で実行可能，MSIL コンパイラを搭載しているので，コンパイルで更なる高速化が可能，プラグインで FeliCa や Twitter との連携機能を利用可能，C# を使って独自のプラグイン制作が可能など実用的である。このような事から，プロデルを業務用途で利用するのに適した環境は整っているように見える。しかしながら，プロデルは構文を自然言語の日本語に近づけて作られた言語であり，それを利用すると考えられる層は，アンケート結果より，プログラミング未経験の初学者や，プログラミング言語の学習途中の学生などの層である。

このように，教育分野での普及，業務分野での普及ともに，利用するのに適した環境は整っているように見える。しかしながら，構文が自然言語の日本語に近づけて作られた日本語プログラミング言語である「プロデル」は，構文の性質が日本語に近い分，プログラミング初学者に利用されやすいが，業務分野で利用するのに適した環境になっている。すなわち，プログラミング初学者に利用されやすい構文であるが，教育分野では利用されていないなど，日本語プログラミング言語の構文の性質と，利用されている環境が「ミスマッチ」を起こしている。そして，そのような「ミスマッチ」を起こさず，日本語プログラミング言語の普及率をより上げるには，プログラミング教育が 2020 年度から小学校，2021 年度から中学校，2022 年度から高校で必修化

されたことを利用し，初等教育でのプログラミング教育でプログラミング的思考が育まれた状態にし，中等教育以降でのプログラミング教育で日本語プログラミング言語を活用することが重要である。

4.5. プログラミング教育での活用

はじめに，プログラミング教育の目的はプログラミング的思考を養うことにあり，プログラミング的思考とは，「自分が意図する一連の活動を実現するために，どのような動きの組合せが必要であり，一つ一つの動きに対応した記号を，どのように組み合わせたらいいのか，記号の組合せをどのように改善していけば，より意図した活動に近づくのか，といったことを論理的に考えていく力」と文部科学省によって定義されている。また，「児童がおのずとプログラミング言語を覚えたり，プログラミングの技能を習得したりするといったことは考えられますが，それ自体をねらいとしているのではない」とも述べられている。つまり，プログラミング教育の目的は，技能の習得ではなくプログラミング的（論理的）思考力を身に着けることである。

そして，そのような目的であるプログラミング教育において日本語プログラミング言語を活用していくには，「動きに対応した記号」が日本語で表現されている日本語プログラミング言語で書かれたコードは，日本語を理解できれば手順書のように読めるが，曖昧さを含まないように一定のルールに従って記述されている必要がある。この特性を活かして，読解力やわかりやすく記述する力を養うといった活用ができるのではないかと考える。

4.6. 初等教育からのプログラミング教育の必要性

プログラミング教育の必要性として最も重要な要素に，論理的思考力を鍛えることがある。そして，その論理的思考力を早期段階である初等教育の頃から鍛えることが重要であると考える。また，文部科学省の文書には，「今後の社会の在り方について，とりわけ最近では，『第4次産業革命』ともいわれる，進化した人工知能が様々な判断を行ったり，身近な物の働きがインターネット経由で最適化されたりする時代の到来が，社会の在り方を大きく変えていくとの予測がなされているところである。教育界には，そのような社会的変化の中でも，子供たちが自信を持って自分の人生を切り拓き，よりよい社会を創り出していくことができるそうした資質・能力として，読解力，論理的思考力，創造性，問題解決能力などは，時代を超えて常にその重要性が指摘されてきており，これからの時代においてもその重要性が変わることはない。

これらに加えて，情報や情報技術を問題の発見・解決に活用していく力（情報活用能力）の重要性も高まっている。学校教育は，こうした資質・能力の育成に向けて充実を図らなければならない。」というような記述もあり，これからの時代，より一層重要性が増すと考える。

また，第 4.4 節，普及率が低い原因でも述べたように，日本語プログラミング言語の普及率をより上げるには，プログラミング教育が 2020 年度から小学校，2021 年度から中学校，2022 年度から高校で必修化されたことを利用し，初等教育でのプログラミング教育でプログラミング的思考が育まれた状態にし，中等教育以降でのプログラミング教育で日本語プログラミング言語を活用することが重要である。また，この方法が日本人にとっては最も学習コストが低く，効率の良い学び方だと考える。そのため，中等教育以降での学習成果をより上げるためにも，初等教育でのプログラミング教育による下積みが必要性として挙げられる。

4.7. プログラミング教育の実態

コロナウイルスの影響による大きな遅延はあったものの，2020 年度から初等教育でのプログラミング教育が実際に開始された。そこで，プログラミング教育の実態調査を行い，現場の教師はどのような実感や考えであるか，小学校教師の方に「児童の読解力・記述力は 5 年ほど前と比較してどう変わったか」，「児童の算数の回答力は 5 年ほど前と比較してどう変わったか」，「小中学校でのプログラミング教育を必要と感じるか否か」，「現在のプログラミング教育について困っていること」，「日本語で記述できるプログラミング言語を初等教育に利用してみたいか否か」の 5 項目についてご回答頂いた。

まず初めに，プログラミング教育で向上するといわれている読解力や記述力の現状について把握するため，「児童の読解力・記述力は 5 年ほど前と比較してどう変わったか」について述べる。児童の読解力・記述力については，低下の傾向にあるそうである。それは，日頃から自分の頭で考えることが減ってきていることと関係しているように感じ，答えがわからなかったらすぐに大人に聞いたり，調べたりする習慣がつき，考える力を養う機会を逃しているように思えるとのことである。その結果，文章から様々なことを読み取ったり，想像したりすることが苦手な子たちが増え，また，記述力に関しても，「何を書いたらいいかわからない」と考えることをやめてしまう子や，頭の中で自分の思考を整理して文章に表すことが苦手な子が増えてきている現状であるそうだ。低下の一途をたどっているからこそ，このタイミングで初等教育でのプログラミング教育が開始され

たのは正解であったと言える。

次に，プログラミング教育で能力を育んでいく際に重要な部分となっていく算数の力について把握するため，「児童の算数の回答力は5年ほど前と比較してどう変わったか」について考察する。児童の算数の回答力基本的な計算の力は大きな変化はないように感じるそうである。しかし，子どもが自分の考えを発表する場面で，友達にわかりやすく，順序立てて説明することを苦手とする子は増えてきているようである。プログラミング的思考とは，「自分が意図する一連の活動を実現するために，どのような動きの組合せが必要であり，一つ一つの動きに対応した記号を，どのように組み合わせたらいいのか，記号の組合せをどのように改善していけば，より意図した活動に近づくのか，といったことを論理的に考えていく力」と文部科学省によって定義されているが，まさにこの力が低下してきているということがわかる。

次に，「小中学校でのプログラミング教育を必要と感じるか否か」について考える。論理的思考力を鍛えることは今の子たちにとって大切なことだと思い，従来の国語や算数などの科目の中でも，教材の扱い方や教師側の指導の仕方によって論理的思考力を育てることができるのではないかとのことである。これは，2020年度から開始された初等教育でのプログラミング教育の手法である，既存の科目にプログラミングの要素を取り込むという方法でやっていけるのではないかということである。

次に，初等教育でのプログラミング教育は2020年度より始まったばかりのため，なにかしらトラブルや解決しなければならない事柄はあるのか把握するため，「現在のプログラミング教育について困っていること」についてまとめる。初等教育でのプログラミング教育の開始1年目時点では，プログラミング教育が算数や理科などの教科書の中で一部扱われるようになってきたものの，国や県，市の研修などが進んでおらず，何をどのように教えていけばよいのか個人に任されている部分が多いため，はっきりとわかっていないのが現状のようである。これに関しては，開始1年目だからという事も考えられるが，最も大きな要因としては，コロナウイルスの流行により，2020年度の授業や研修が大幅に遅延したことが挙げられると考える。

最後に，「日本語で記述できるプログラミング言語を初等教育に利用してみたいか否か」について述べる。日本語で記述できるプログラミング教材は小学校の高学年，中学生くらいであれば活用できるのかも知れないが，まず，初等教育では国語や算数などの教科学習の中で，論理的思考力を育てていくことが大切であると考えているとのことである。やはり初等教育の間は，現状のプログラミング

教育のやり方でプログラミング的（論理的）思考力を育んでいくことが大切であり，日本語プログラミング言語などを導入するならば，中等教育以降で行うことが適切であると考える。

5. まとめ

本稿では，日本語プログラミング言語の普及を妨げる原因について，英語記述のプログラミング言語との可読性の比較，アンケート調査などを踏まえて考察し，普及方法や新たな活用方法を模索した。

日本語プログラミング言語の中では，ドリトルが，なでしこやプロデルと比べて最も教育分野に特化しており，教育現場を中心に普及が進んでいるのが特徴である。リファレンスやマニュアルも主には授業用テキストや教師向けの授業資料という形で整備され，配布されている。このような事から，ドリトルを教育分野で利用するのに適した環境は整っているように見える。しかしながら，ドリトルは構文を英語記述のプログラミング言語に近づけて作られた言語であり，それを利用すると考えられる層は，アンケート結果より，いずれかのプログラミング言語を習得している，もしくはプログラミング言語の学習過程にあるが，すでにある程度十分な知識を持っている層である。

日本語プログラミング言語の中では，プロデルが，なでしこやドリトルと比べて最も業務分野に特化している。プロデルは，日本語プログラミング言語の中では珍しく，教育分野での使用のためには作られておらず，汎用的なソフトウェア開発を目標としているのが特徴である。バッチ処理や画像描画，GUI 部品の機能が用意されている，Excel, CSV 形式操作，ODBC 接続が可能，ツールやゲーム作りが可能，大量のデータ処理を短時間で実行可能，MSIL コンパイラを搭載しているので，コンパイルで更なる高速化が可能，プラグインで FeliCa や Twitter との連携機能を利用可能，C# を使って独自のプラグイン制作が可能など実用的である。このような事から，プロデルを業務用途で利用するのに適した環境は整っているように見える。しかしながら，プロデルは構文を自然言語の日本語に近づけて作られた言語であり，それを利用すると考えられる層は，アンケート結果より，プログラミング未経験の初学者や，プログラミング言語の学習途中の学生などの層である。

このように，教育分野での普及，業務分野での普及ともに，利用するのに適した環境は整っているように見える。しかしながら，構文が自然言語の日本語に近づけて作られた日本語プログラミング言語である「プロデル」は，構文の性質が日本語に近い分，プログラミング初学者

に利用されやすいが，業務分野で利用するのに適した環境になっている。すなわち，プログラミング初学者に利用されやすい構文であるが，教育分野では利用されていないなど，日本語プログラミング言語の構文の性質と，利用されている環境が「ミスマッチ」を起こしている。そして，そのような「ミスマッチ」を起こさず，日本語プログラミング言語の普及率をより上げるには，プログラミング教育が 2020 年度から小学校，2021 年度から中学校，2022 年度から高校で必修化されたことを利用し，初等教育でのプログラミング教育でプログラミング的思考が育まれた状態にし，中等教育以降でのプログラミング教育で日本語プログラミング言語を活用することが重要である。

参考文献

1）クジラ飛行机,“プログラミング言語大全”，株式会社技術評論社，(2020).

2）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の育成とプログラミング教育に関する有識者会議”，2016 年 4 月 19 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/index.htm［アクセス日：2022 年 11 月 18 日].

3）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の育成とプログラミング教育に関する有識者会議について”,2016 年 4 月 19 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1370404.htm［アクセス日：2022 年 11 月 18 日].

4）文部科学省,“小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ)”，2016 年 6 月 16 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1372525.htm［アクセス日：2022 年 11 月 18 日].

5）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の育成とプログラミング教育に関する有識者会議　議事要旨・議事録・配付資料”，2016 年 6 月 3 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/giji_list/index.htm［アクセス日：2022 年 11 月 18 日].

6）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の育成とプログラミング教育に関する有識者会議（第 3 回）議事録”，2016 年 6 月 3 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/gijiroku/1382219.htm［アクセス日：2022 年 11 月 18 日].

7）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の

育成とプログラミング教育に関する有識者会議（第3回）配付資料”，2016年6月3日更新.
https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/shiryo/1371637.htm［アクセス日：2022年11月18日].

8）AI（人工知能）関連メディア,“【入門編】自然言語処理（NLP）とは｜意味・仕組み・活用例・課題”，2019年11月20日更新.
https://ledge.ai/nlp/［アクセス日：2022年11月18日].

9）Hatada’sHomePage（旧),“プログラミング言語の基本概念”，2016年更新.
http://shopping2.gmobb.jp/htdmnr/www08/lp2016/chap01/language.html［アクセス日：2022年11月18日].

10）IT用語辞典,“機械語”，2018年10月24日更新.
https://e-words.jp/w/%E6%A9%9F%E6%A2%B0%E8%AA%9E.html［アクセス日：2022年11月18日].

11）HPCS Lab.,“言語処理系とは”，2014年更新.
http://www.hpcs.cs.tsukuba.ac.jp/~msato/lecture-note/comp-lecture/note1.html［アクセス日：2022年11月18日].

12）TECHのススメ,“プログラミング言語とは？”，2020年9月16日更新.
https://haa.athuman.com/media/it/programming/2120/［アクセス日：2022年11月18日].

13）TechTarget,“Javaから「Kotlin」に乗り換えたくなる5つの理由”，2020年2月17日更新.
https://techtarget.itmedia.co.jp/tt/news/2002/17/news06.html［アクセス日：2022年11月18日].

14）なでしこ,“なでしこ3–日本語プログラミング言語”，2022年10月9日更新.
https://nadesi.com/doc3/［アクセス日：2022年11月18日].

15）日本語プログラミング言語「プロデル」,“プロデル”，2022年11月12日更新.
https://rdr.utopiat.net/［アクセス日：2022年11月18日].

16）プログラミング言語「ドリトル」,“教育用プログラミング言語「ドリトル」情報ページ”，2021年11月13日更新.
https://dolittle.eplang.jp/［アクセス日：2022年11月18日].

17）TECH CAMP,“「なでしこ」「プロデル」「ドリトル」三大日本語プログラミング言語を解説”，2021年2月1日更新.
https://tech-camp.in/note/technology/41396/#i-9［アクセス日：2022年11月18日].

18）ACE PRO,“衝撃の事実”，2022年2月22日更新.
https://www.ace-schools.co.jp［アクセス日：2022年11月18日].

1. ICT 委員会　会議報告

愛知大学情報メディアセンターの事業および運営は，ICT 企画会議のもと，三校舎合同のICT 委員会を設置し，豊橋および名古屋(車道メディアゾーン含む)情報メディアセンターの事業を推進する。　（2021 年 10 月から 2022 年 9 月まで）

2021 年度
◇第 3 回 12 月 2 日
議題：
1. 2022 年度予算申請について
2. 2022 年度情報メディアセンター開館カレンダーについて
3. その他

協議・報告：
1. 学生推奨 PC について
2. その他

2022 年度
◇第 1 回 6 月 30 日
議題：
1. 所長改選について
2. COM 編集委員選出について
3. 豊橋校舎　新棟2Fのレイアウトについて
4. その他

協議・報告
1. 2021 年度事業報告書について
2. 2022 年度事業計画書について
3. その他

◇第 2 回 9 月 29 日
議題：
1. 所長改選について
2. 補正予算申請について
3. その他

協議・報告：
1. 2023 年度実習室アンケートについて
2. 2023 年度予算申請について
3. 学生の推奨ノートパソコンの環境について
4. その他

2. 情報メディアセンター主催行事
（2021 年 10 月～ 2022 年 9 月）

◆名古屋校舎

開　講　日	講　習　会　名	教室	参加人数
2021年10月22日（金）	Word 講習会・卒論編	L707	3 人
2021年11月17日（水）	Excel 講習会・グラフ編	L713	3 人
2021年12月17日（金）	Excel 講習会・関数編	L711	6 人
2022年 4 月 6 日（水）	新入生向けパソコン利用ガイダンス	W402	41 人
2022年 4 月 6 日（水）	新入生向けパソコン利用ガイダンス	W402	14 人
2022年 4 月 7 日（木）	新入生向けパソコン利用ガイダンス	W402	19 人
2022年 4 月19日（火）	PowerPoint 講習会	W401	36 人
2022年 5 月12日（木）	PowerPoint 講習会	L711	4 人
2022年 5 月18日（水）	Word 講習会・レポート編	L711	4 人
2022年 6 月16日（木）	Word 講習会・レポート編	L711	3 人
2022年 6 月20日（月）	PowerPoint 講習会	L707	15 人
2022年 6 月27日（月）	Word 講習会・レポート編	L712	2 人
2022年 7 月12日（火）	Word 講習会・卒論編	オンライン	9 人
2022年 9 月28日（水）	Word 講習会・卒論編	L711	4 人

◆豊橋校舎

開　講　日	講　習　会　名	教室	参加人数
2022年 6 月24日（金）	PowerPoint 講習会（PowerPoint2016）	523 教室	4 人
2022年 6 月24日（金）	Excel 講習会（Excel2016）	413 教室	5 人
2022年 6 月29日（水）	Word 講習会（Word2016）	413 教室	5 人

◆車道校舎：主催行事なし

2021 年度 Moodle（LMS）運営業務報告

1. Moodle 講習会

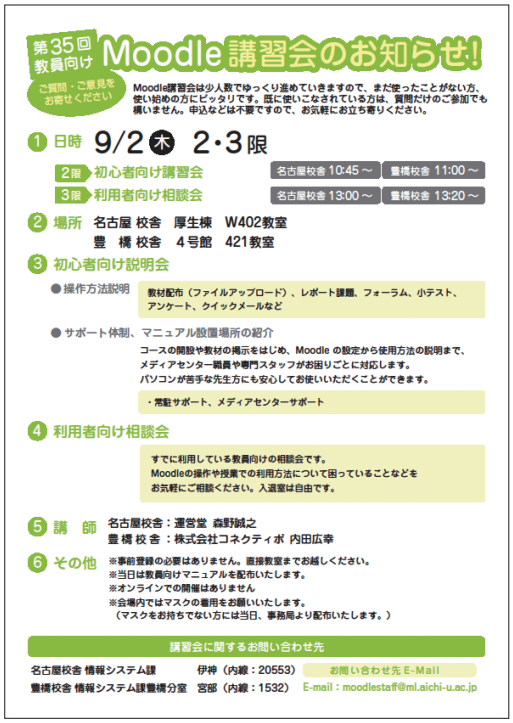
Moodle の利用促進および遠隔授業サポートのため, Moodle 講習会を以下の通り実施した。

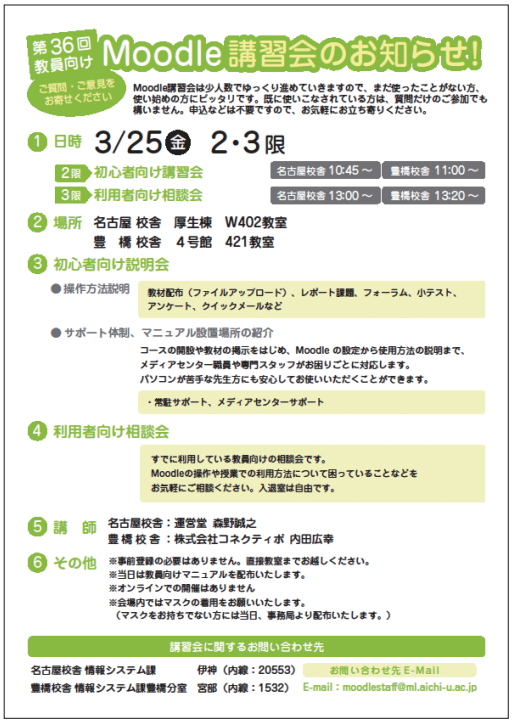
① 第 35 回 Moodle 講習会

校舎	開催日時	場所
豊橋	2021 年 9 月 2 日	4 号館 421 教室
名古屋	2・3 限	厚生棟 W402 教室

② 第 36 回 Moodle 講習会

校舎	開催日時	場所
豊橋	2022 年 3 月 25 日	4 号館 421 教室
名古屋	2・3 限	厚生棟 W401 教室





2. Moodle 利用状況

（A）コース利用状況

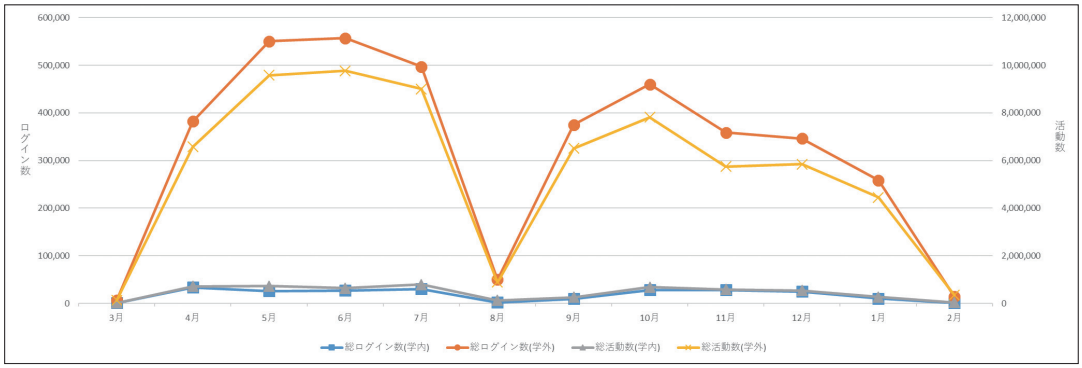
運用開始 13 年目の 2021 年度は，新型コロナウイルス感染拡大防止のため，2020 年度に続き全ての授業についてコース作成をすることとなった。

カテゴリ	2021 年度　利用コース数			
	春学期	秋学期	通年	合計
共通教育科目（名古屋）〈法・経済・経営・現中・国際〉	433	427	0	860
共通教育科目（豊橋）〈文・地域・短大〉	267	256	1	524
法学部	70	76	39	185
経済学部	76	65	75	216
経営学部	114	142	74	330
現代中国学部	174	129	23	326
国際コミュニケーション学部	176	199	5	380
文学部	226	214	16	456
地域政策学部	138	144	8	290
短期大学部	72	75	3	150
大学院	98	93	93	284
法科大学院	50	48	1	99
資格課程	84	93	18	195
協定留学生日本語コース	0	0	0	0
自習用教材	0	0	35	35
ヘルプ	0	0	7	7
その他	0	0	17	17
合計	1978	1961	415	4354

（B）サイトアクセス状況

2021 年度は 2020 年度に続いてコロナ禍で Moodle が運用された。2020 年度は遠隔授業による Moodle での運用が開始されたのが 5 月ゴールデンウィーク以降であったため，前年同月比では 3 月と 4 月の利用者が大きくなっているが，それ以降の利用者は昨年と同様に推移している。

2021 年度　学内・学外からのログイン数・活動数推移 (月別)



		3 月	4 月	5 月	6 月	7 月	8 月	9 月	10 月	11 月	12 月	1 月	2 月	合計	平均
2019 年度	総ログイン数（学内）	421	23,767	34,861	32,118	37,200	990	17,824	34,205	28,272	27,356	14,593	347	251,954	20,996
	総ログイン数（学外）	1,276	38,789	58,230	53,939	74,926	7,418	26,245	48,612	44,312	44,681	49,042	2,216	449,686	37,474
	総活動数（学内）	22,626	461,465	574,161	533,844	700,879	19,288	276,896	471,716	357,383	375,973	239,322	7,007	4,040,560	336,713
	総活動数（学外）	26,069	505,463	746,053	766,831	1,164,778	105,650	306,291	544,806	483,347	528,962	650,874	42,804	5,871,928	489,327
	ログインあたり活動数（学内）	53.74	19.42	16.47	16.62	18.84	19.48	15.54	13.79	12.64	13.74	16.4	20.19	16.04	19.74
	ログインあたり活動数（学外）	20.43	13.03	12.81	14.22	15.55	14.24	11.67	11.21	10.91	11.84	13.27	19.32	13.06	14.04
2020 年度	総ログイン数（学内）	122	2,713	8,487	6,698	5,394	2,123	7,793	32,979	27,522	26,722	9,697	892	131,142	10,929
	総ログイン数（学外）	1,272	20,006	648,209	661,396	685,698	139,800	357,397	465,944	363,780	353,502	272,696	16,835	3,986,535	332,211
	総活動数（学内）	1,704	136,249	523,016	254,279	201,620	163,036	427,351	707,450	577,629	560,515	236,015	51,181	3,840,045	320,004
	総活動数（学外）	25,766	289,722	13,123,453	13,463,238	14,148,403	2,907,150	6,517,112	8,293,424	6,204,644	6,213,003	4,806,766	357,769	76,350,450	6,362,538
	ログインあたり活動数（学内）	13.97	50.22	61.63	37.96	37.38	76.8	54.84	21.45	20.99	20.98	24.34	57.38	29.28	39.83
	ログインあたり活動数（学外）	20.26	14.48	20.25	20.36	20.63	20.8	18.23	17.8	17.06	17.58	17.63	21.25	19.15	18.86
2021 年度	総ログイン数（学内）	769	33,432	25,885	26,803	30,282	1,781	9,686	28,122	28,240	25,015	9,977	761	220,753	18,396
	総ログイン数（学外）	6,384	381,959	550,583	557,142	497,146	49,916	374,894	460,258	358,565	346,202	258,510	14,112	3,855,671	321,306
	総活動数（学内）	12,383	711,677	719,977	632,517	785,147	120,347	251,019	686,218	581,506	543,247	266,535	33,871	5,344,444	445,370
	総活動数（学外）	126,897	6,578,419	9,580,164	9,771,776	9,002,582	887,609	6,510,506	7,812,320	5,742,774	5,842,780	4,449,869	347,558	66,653,254	5,554,438
	ログインあたり活動数（学内）	16.1	21.29	27.81	23.6	25.93	67.57	25.92	24.4	20.59	21.72	26.71	44.51	24.21	28.85
	ログインあたり活動数（学外）	19.88	17.22	17.4	17.54	18.11	17.78	17.37	16.97	16.02	16.88	17.21	24.63	17.29	18.08
前年同月比	総ログイン数（学内）	630.30%	1232.30%	305.00%	400.20%	561.40%	83.90%	124.30%	85.30%	102.60%	93.60%	102.90%	85.30%	168.30%	168.30%
	総ログイン数（学外）	501.90%	1909.20%	84.90%	84.20%	72.50%	35.70%	104.90%	98.80%	98.60%	97.90%	94.80%	83.80%	96.70%	96.70%
	総活動数（学内）	726.70%	522.30%	137.70%	248.70%	389.40%	73.80%	58.70%	97.00%	100.70%	96.90%	112.90%	66.20%	139.20%	139.20%
	総活動数（学外）	492.50%	2270.60%	73.00%	72.60%	63.60%	30.50%	99.90%	94.20%	92.60%	94.00%	92.60%	97.10%	87.30%	87.30%

3. ICT 委員会構成員

◆ICT 委員（2022 年 10 月 1 日現在）

役職名	所　　属	氏　　名
情報メディアセンター所長	経　営　学　部	岩田　員典
委　　員	文　　学　　部	近藤　暁夫
	地 域 政 策 学 部	蒋　　湧
	短 期 大 学 部	迫田　耕作
	法　　学　　部	松井　吉光
	経　営　学　部	毛利　元昭
	現 代 中 国 学 部	吉川　　剛
	経　済　学　部	池森　　均
	国際コミュニケーション学部	梅垣　敦紀
	法 科 大 学 院	春日　　修

◆情報メディアセンター事務室

情報 システム課	課　　長	秦　俊一郎
	係　　長	石原有希子
		水谷　伸司
	課　　員	伊神　真悟
		岩田　大輝
情報システム課 豊橋分室	係　　長	宮部　浩之

4. 愛知大学　情報メディアセンター沿革・歴代所長

年度	組織		所長（任期）		システム沿革
			豊橋	名古屋	
1978					IBM 製ホストコンピュータ 4331　導入
1979					
1980	電子計算機センター	電子計算機センター委員会	津村　善郎 （1980. 4. 1～1982. 4.30）		
1981					
1982					
1983			福田　治郎 （1982. 5. 1～1985. 3.31）		
1984					
1985			高橋　正 （1985. 4. 1～1989. 3.31）		
1986					
1987					
1988					第 1 期教育研究情報システム稼動　1988.4-1991.3
1989	情報処理センター	情報処理センター委員会 豊橋情報処理センター委員会 名古屋情報処理センター委員会	藤田　佳久 （1989. 4. 1～1994. 9.30）	坂東　昌子 （1989. 4. 1～1990. 9.30）	日立製ホストコンピュータ（HITAC M-640/20）導入
1990					
1991				浅野　俊夫 （1990.10. 1～1992. 9.30）	第 2 期教育研究情報システム稼動　1991.4-1994.3
1992					
1993				有澤　健治 （1992.10. 1～1994. 9.30）	
1994					第 3 期教育研究情報システム稼動　1994.10-1997.3 （全校舎学内 LAN 敷設）
1995			樋口　義治 （1994.10. 1～1998. 9.30）	長谷部　勝也 （1994.10. 1～1998. 9.30）	
1996					
1997					第 4 期教育研究情報システム稼動　1997.4-2000.9 （延長 6 ヶ月）
1998			宮沢　哲男 （1998.10. 1～2000. 3.31）	有澤　健治 （1998.10. 1～2000. 9.30）	
1999					
2000			小津　秀晴 （2000. 4. 1～2002. 9.30）	田川　光照 （2000.10. 1～2002. 9.30）	10 月　第 5 期教育研究情報システム稼動
2001					
2002			龍　昌治 （2002.10. 1～2008. 9.30）		
2003				坂東　昌子 （2002.10. 1～2006. 9.30）	
2004	情報メディアセンター	情報メディアセンター委員会 豊橋情報メディアセンター委員会 名古屋情報メディアセンター委員会			4 月　第 6 期教育研究情報システム稼動
2005					
2006		情報メディアセンター運営会議 豊橋情報メディアセンター運営会議			
2007				中尾　浩 （2006.10. 1～2008. 9.30）	
2008		ICT 企画会議 豊橋 ICT 委員会			4 月　第 7 期教育研究情報システム稼動
2009			蒋　湧 （2008.10. 1～2010. 9.30）	伊藤　博文 （2008.10. 1～2012. 9.30）	
2010					
2011			沓掛　俊夫 （2010.10. 1～2012. 9.30）		
2012		ICT 委員会			4 月　新名古屋校舎システム稼働
2013			中尾　浩 （2012.10.1～2014.9.30）		
2014					
2015			松井　吉光 （2014.10. 1～2016. 9.30）		
2016					
2017			松井　吉光 （2016.10. 1～2018. 9.30）		
2018			岩田　員典 （2018.10. 1～）		
2019					
2020			岩田　員典 （2020.10. 1～）		
2021					
2022			岩田　員典 （2022.10.1～）		

編集後記

今回は4本の論文を掲載する形で，COM第47号を発刊する運びとなりました。取り上げられた内容は，AIエージェント，GISデータ解析，IoTリモートセンシング，プログラミング教育と，いずれもホットな話題でバラエティに富んでおりました。また，開発されたツール・手法などを利用しやすいよう丁寧に説明されておりました。これらの研究活動が愛知大学で行われたことを嬉しく誇らしく思うとともに，寄稿いただいた著者の方々に深い感謝の念を抱いております。

AIと言えば，最近は人間の創作物に対する機械学習技術が大きく進歩しているのか，画像，文章，楽曲などを，指定したキーワードやパラメータから自動生成するサービスが次々と現れ，あるいはそれらを活用した作品が世の中に出回り始めました。未成熟な部分も時折見られはしますが，そのままでも人間の仕事と区別がつかない物も多く，野暮で不器用な私は驚いてばかりです。少し前までは「創作活動はAIに代替されない」がある種の通説でしたが，強烈なクサビが打ち込まれているように思います。創作の界隈では，権利関係などで大いに問題視する方々もいらっしゃれば，自分自身のクローンAIとセッションするなど積極的に楽しむ方々もいらっしゃると聞きます。今後が大変気になるところです。

リモートセンシングと言えば，コロナ禍の影響か，いたるところで非接触の計測がなされています。感染拡大防止のために様々な制約を受けている中，人間の行動を大きく制限しない技術と工夫に助けられていることを実感しています。その制約も徐々に緩んできており，例えば今年度の本学では多くの授業が対面でなされ，来年度はほぼ全てがそうなることが期待されています。このように，警戒を必要としつつも人流が増えることから，人間の邪魔をしないリモートセンシングがますます活躍すると思われます。体調管理という意味では，接触型ではありますが，活動量計を含むウェアラブルデバイスもますます普及するでしょうか。研究を兼ね，私も一つ持っておきたいです。

情報技術は様々な物事に絡んでおり，また日進月歩で話題は尽きません。愛知大学においても，分野を問わず様々な方々が利活用し，また興味・関心を持たれていることと思います。その情報源や交流の場として本誌をご活用いただく方々が増えましたら幸いです。寄稿・閲覧など形を問わず，お待ちしております。

（M. M.）

自己紹介

情報システム課　課長　秦　俊一郎

2022年度の人事異動で情報システム課長を拝命しました。2014年度に教務部門に異動して以来、約10年ぶりの情報部門となりました。古巣の職場に戻ったと言えばそうですが、10年も空いてしまうと全てのことが新鮮で、一から再スタートという気持ちです。昨年度まではいちユーザーとして情報システムを利用してきましたが、再び管理者として関わることになり身が引き締まる思いです。教務部門に在籍していた時には新型コロナウィスル感染症への対応にあたりましたが、情報部門に在籍していた経験を多少ではありますが生かすことができたように、これからはユーザー部門での経験をシステム管理者として生かしていければと考えています。大学における情報システムの位置付けとその管理にあたる情報システム課の役割、責任の大きさを考えると大きな重圧を感じますが、安全性、利便性、採算性のバランスを取りながら、本学学生・教職員の活動に寄与する情報システムの企画・導入・運用に努めてまいる所存です。皆様におかれましては、ご指導いただければ幸いに存じます。どうぞよろしくお願いいたします。

自己紹介

情報システム課　岩田　大輝

2012年度に愛知大学へ入職し，学生課にて5年，入試課にて5年の勤務を経て，2022年度から情報システム課へ異動となりました。現在は事務情報システム，ユーザーアカウントの管理等を担当しております。

学生課では主に奨学金業務を担当し，入試課では入試広報部門でオープンキャンパスの企画運営，大学案内の制作，進路ガイダンス等の業務を担当してまいりました。情報システム部門については全くの初めての分野であり，戸惑いが多い毎日である一方，新しい知識や経験に触れる事ができ，ワクワクしております。

学生課に在籍していたころは，主に窓口業務が中心であり，学生一人一人が抱える事情も様々であったため，学生と密なコミュニケーションを取りながら，個々の現状にあった最適解を導き出す事に注力しておりました。

入試課に在籍していたころには，愛知大学の魅力をいかに的確に，高校生や保護者に伝えるか。どのような情報を発信すれば高校生が魅力的に感じてくれるのか。という事を常に考えながら仕事に取り組んでおりました。

情報システム課は，私がこれまで経験してきた分野とは異なりますが，いつか何かしらの形で私のこれまでの経験である点が線に繋がる事を信じて，邁進してまいります。

私の趣味についても少し触れさせていただきます。春から秋にかけては，良く登山やキャンプに出掛けております。少しでも長い時間，山で過ごせるように。と数年前に実家がある三重県の北部へ帰郷し，早朝から山行に出掛ける日々を送っております。道中では野生のリスが木の葉で遊んでいたり，鹿がひなたぼっこをしていたりと，都会では味わう事ができない非日常を楽しめます。

冬場は思うように山行に行けない事もあり，珈琲豆の焙煎をしております。珈琲豆に関しては非常に奥深いものがあり，ここでは語りつくせないのですが，当日の天候，気温，湿度等の細かな条件によって，風味やコクが変化します。そのため30秒毎に窯内の温度の変化，ガス圧，ダンパー（排気）を調整しながら，好みの味を探っていきます。焙煎した珈琲豆に関しては，近所の方から野菜や果物をいただいた際のお返しとして，お渡しするなどして楽しんでおります。

私の趣味が情報システム課のお仕事とつながる事は少ないかと思いますが，趣味と同じぐらい没頭して，業務にも取り組みたいと思っております。

以上，長々と拙い文章にお付き合いいただき有難うございました。精一杯，楽しみながら頑張りたいと思います。今後ともよろしくお願いいたします。

2020 年 7 月 1 日改訂

愛知大学情報メディアセンター紀要〈COM〉　原稿募集要項

情報メディアセンター紀要〈COM〉は，下記の要領で原稿を募集しています。詳細につきましては，情報メディアセンターまでお問い合わせください。

1. 著者の資格

(1) 本学教職員および本学教職員との共著者
(2) 本学非常勤教員
(3) 本学学生（教員と共著とする。）
(4) 編集委員会が認めたもの

2. 投稿原稿の内容

投稿原稿は未発表のもので，下記に関係する内容とする。
(1) 情報教育に関する理論と実践
(2) 情報科学や情報工学に関する理論とその応用
(3) 情報システムに関する調査，分析，理論
(4) コンピュータを活用した研究，教育，および業務等の実践報告
(5) 本学のコンピュータ利用に関して必要と思われる情報メディアセンターの報告
(6) その他（編集委員会が認めたもの）

3. 投稿原稿の区分

投稿された原稿は編集委員会の審査に従って，下記のように区分して掲載する。ただし，法令等に抵触する，内容に著しい不備がある，執筆要項に従わないなどの問題があるものは，原稿の修正を依頼することや，掲載を見あわせることがある。
(1) 論文
(2) 研究ノート
(3) 情報教育実践報告
(4) 書評（新刊・古典・ソフトウェア）
(5) 学会動向
※原稿の体裁と見本については別紙を参照のこと。

4. 原稿の提出要領

(1) 原稿は，電子ファイルで提出すること。
(2) 完成された投稿原稿のみを受理する。
(3) 提出する電子ファイル名は，投稿原稿のタイトルとすること。
(4) 図版等がある場合は，その電子ファイルもあわせて提出すること。
図版等のファイル形式は jpeg，pdf とする。
(5) 提出ファイルは，原則 Microsoft Word またはテキスト形式とする。

ただし，その他の形式であっても編集委員会が認めた場合は受理する。

(6) 裏表紙（目次用）として，タイトル，著者名の欧文を添えること。

(7) 著者は連絡先（ゲラ等の送付先）の住所，電話番号を申し込み先の担当者まで連絡すること。

5. 投稿原稿の体裁

投稿原稿は横書きとし，図・表などは適切な場所に分かりやすく挿入すること。なお，投稿原稿はCOM編集委員会にて共通したフォーマットに統一する。

6. 校正

(1) 校正は著者校正を2回とする。

(2) 校正段階での内容の変更は，編集作業に支障をきたさない範囲で行なうこと。

7. 著作権

(1) 提出された論文の著作権は，原則として愛知大学情報メディアセンターに属し，無断で複製あるいは転載することを禁じる。

(2) 論文作成に際して用いたコンピュータソフトや映像ソフト等の著作権に関する問題は，著者の責任において処理済みであること。他人の著作権の侵害，名誉毀損，その他の問題が生じないよう十分に配慮すること。

(3) 万一，執筆内容が第三者の著作権を侵害するなどの指摘がなされ，第三者に損害を与えた場合，著者がその責を負う。

(4) 著作人格権は著者に属する。

(5) 本誌に掲載された原稿は，学内においては，愛知大学情報メディアセンターホームページおよび愛知大学リポジトリにてデジタル公開するものとする。

(6) 本誌に掲載された原稿は，学外においては国立情報学研究所等へ登録される。

8. その他

(1) 別刷りは著者に対して希望を調査し，30部を上限として無料進呈する。

(2) 著者には紀要を2部進呈する。ただし希望があれば10部を限度として進呈する。

以上

申し込み・問い合わせ：愛知大学情報メディアセンター
担当：情報システム課　岩田　大輝
E-mail：johosystem@ml.aichi-u.ac.jp
TEL：052-564-6117（内線 20554）
FAX：052-564-6217（内線 20569）

2014 年 8 月 8 日改訂

愛知大学情報メディアセンター紀要〈COM〉　執筆要項

1. 執筆言語

和文もしくは英文とする。

2. 原稿

(1) **論文**……和文の場合は 30,000 文字程度，英文の場合は 15,000 words 程度を上限とする。ただし，図版等の数量に応じて調節すること。

(2) **研究ノート**……和文の場合は 20,000 文字程度，英文の場合は 10,000 words 程度を上限とする。　ただし，図版等の数量に応じて調節すること。

(3) **情報教育実践報告**……和文の場合は 20,000 文字程度，英文の場合は 10,000 words　程度を上限とする。ただし，図版等の数量に応じて調節すること。

(4) **書評（新刊・古典・ソフトウェア）**・・・和文の場合は 5,000 文字程度，英文の場合は 3,000 words 程度を上限とする。書評（新刊・古典）には図版等を挿入することはできないが，ソフトウェアレビューについては若干の図版を添えることが出来る。

(5) **学会動向**……COM のフォーマットに従う。
長文の場合は分裁や再提出等の措置を求めることがある。

3. 著者と所属

著者名と所属を記載し，著者名のあとにカッコ（　）に入れて所属を記載する。

4. セクションタイトルとセクション記号

本文中の章，節，項，目などの立て方は，原則として以下のとおりとする。

（例）

1. 章タイトル
1.1　節タイトル
1.1.1　項タイトル
(1)　目タイトル

5. 図・表・写真

図・表・写真は，本文中の適当な箇所に挿入すること。または，挿入箇所を明確にすること。

ただし，COM 編集委員会にて挿入位置，サイズを変更する場合があるが，変更不可の場合は明記のこと。

(1) 表について
表の上部に「表○　表名」（○は表の一連番号）を記載すること。

(2) 図・写真について

図・写真の下部に「図○　図名」(○は図の一連番号) または「写真○　写真名」(○は写真の一連番号) を記載すること。

6. 要旨とキーワード

論文と研究ノートには要旨とキーワードをつける。要旨は 400 字以内 (200words 以内) で執筆し，本文と同じ言語でもよいし，異なった言語でもよい。キーワードは国立情報学研究所の CiNii 等への正確な登録のために, 5 ～ 7 語程度のキーワードをつける。

7. 謝辞

謝辞を記載する場合は，本文の最後に謝辞と小見出しを使い記載する。

8. 注

注を記載する場合は，以下のいずれかの方法による。

(1) 該当ページの下部または見開きの前後 2 ページ分の後のページの本文の下部に脚注として記載する。

(2) 本文の末尾に後注として一括して記載する。本文の後に 1 行空けてから「注」という見出しを立て，その次の行から，注を一括して記載する。

上記のいずれの場合も本文中の該当箇所には，番号と右丸括弧を使い注 1) のように上付きで記すこと。

9. 参考文献

参考文献の記載は，本文の後（注がある場合は注の後）に 1 行空けてから「参考文献」という見出しを立て, その次の行から, 参考文献を一括して記載すること。本文中の該当箇所には，番号と右丸括弧を使い 1) のように上付きで記すこと。

参考文献は原則として，雑誌の場合には，著者，標題，雑誌名，巻，号，ページ，発行年を，単行本の場合には，著者，書名，ページ数，発行所，発行年を，この順に記す。引用番号の記し方は本文上に出現した順番とし，次の例を参照にされたい。

(例)

1) 山田太郎：偏微分方程式の数値解法，情報処理，Vol.1, No.1, pp.6-10 (1960).

2) Feldman, J.and Gries, D.: Translater Writing System, Comm. ACM, Vol.11, No.2, pp.77-113 (1968).

3) 大山一夫：電子計算機，p.300，情報出版，東京 (1991).

4) Wilkes, M. V: Time Sharing Computer Systems, p.200, McDonald, New York (1990).

以上

愛知大学情報メディアセンター紀要 COM〔コム〕

Vol.32 No.1　第 47 号

2023 年 3 月 1 日　印刷
2023 年 3 月 5 日　発行

編集　愛知大学情報メディアセンター
「COM」編集委員会

発行　愛知大学情報メディアセンター
（名古屋）名古屋市中村区平池町四丁目 60-6
〒453-8777　TEL（052）564-6117（直通）
FAX（052）564-6217
（豊　橋）豊橋市町畑町1-1
〒441-8522　TEL（0532）47-4124（直通）
FAX（0532）47-4125
（車　道）名古屋市東区筒井二丁目 10-31
〒461-8461　TEL（052）937-8120（直通）
FAX（052）937-8121

印刷　株式会社クイックス

情報メディアセンター教育用パソコン　機種および設置台数

○豊橋校舎

設　置　場　所		機　　種	台数
情報メディアセンター（4号館）	420教室	富士通 ESPRIMO D587/R	34
		富士通 ESPRIMO D587/S	30
	421教室	富士通 ESPRIMO D587/S	52
	423教室	富士通 ESPRIMO D587/S	60
	424教室	東芝 Dynabook B65/HS	40
	413教室	富士通 ESPRIMO D587/R	25
5号館	514教室	東芝 Dynabook B65/HS	20
	523教室	富士通 ESPRIMO D587/R	50
図書館棟1F	メディアゾーン	富士通 ESPRIMO D587/S	40
豊橋　計			352

○名古屋校舎

設　置　場　所		機　　種	台数
厚生棟4F	W401教室	富士通 ESPRIMO D587	60
	W402教室	富士通 ESPRIMO D587	60
	W403教室	富士通 ESPRIMO D587	60
	W404教室	富士通 ESPRIMO D587	60
	メディアゾーン	富士通 ESPRIMO D587	120
講義棟7F	L707教室	富士通 LIFEBOOK A576/P	80
	L708教室	富士通 LIFEBOOK A576/P	80
	L709教室	富士通 LIFEBOOK A576/P	80
	L710教室	富士通 LIFEBOOK A576/P	24
	L711教室	富士通 LIFEBOOK A576/P	24
	L712教室	富士通 LIFEBOOK A576/P	24
	L713教室	富士通 LIFEBOOK A576/P	24
名古屋　計			696

○車道校舎

設　置　場　所	機　　種	台数
K802	富士通 LIFEBOOK A574/M	35
K804	HP ProBook 4540s	50
メディアゾーン	HP Pro 4300 SFF	6
車道　計		91

Journal of Aichi University Media Center

vol.32 No.1

CONTENTS

Aichi University Media Center