

# 卒業論文

視覚と行動の end-to-end 学習により  
経路追従行動をオンラインで模倣する手法の提案  
(目標方向による経路選択機能の追加と検証)

A proposal for an online imitation method of path-tracking  
behavior by end-to-end learning of vision and action  
(Addition of path selection function and verification by target direction)

2023 年 1 月 24 日 提出

指導教員 林原 靖男 教授

千葉工業大学 先進工学部 未来ロボティクス学科

19C1101 藤原 杠



# 概要

## 視覚と行動の end-to-end 学習により 経路追従行動をオンラインで模倣する手法の提案 (目標方向による経路選択機能の追加と検証)

近年, カメラ画像に基づいた自律走行の研究が行われている。本研究室でも, 測域センサを用いた自律移動システムの出力を教師信号として与えることでロボットの経路追従行動をオンラインで模倣する手法を提案している。また, 実験によりカメラ画像に基づいた自律走行で, 一定の経路が周回可能であることを確認している。本研究では, 目標の進行方向をデータセットと学習器の入力に加えることで, 「直進」や「左折」などの経路が選択できる分岐路において, 任意の経路を選択可能にする機能の追加を提案する。提案手法では, 学習時に測域センサを用いた自律移動システムで走行しながら, 教師信号であるシステムの出力とカメラ画像に加えて, 目標方向をデータセットに追加することで模倣学習を行う。学習後, カメラ画像と目標方向に基づいて経路を選択して自律走行を行う。ただし, この学習には多くの時間が必要であることが研究により明らかになった。そのため, 2つのアプローチを行い, 学習時間の短縮を図った。これらの提案を検証するため, シミュレータを用いた実験と実環境での実験を行い, 有効性の検証を行った。

キーワード: end-to-end 学習, ナビゲーション, 目標方向

# abstract

A proposal for an online imitation method of path-tracking behavior by end-to-end learning of vision and action

(Addition and verification of path selection function by target direction)

In recent years, research on autonomous moving based on camera images has been conducted. In this laboratory, we have also proposed a method to imitate the path-following behavior of a robot online by providing the output of an autonomous moving system using a range-finding sensor as a teacher signal. Experiments have also confirmed that a certain path can be circumnavigated by the autonomous moving based on camera images. In this study, we propose the addition of a function that enables the selection of arbitrary paths at branching roads where paths such as "go straight" or "turn left" can be selected by adding the target direction to the dataset and the input of the learner. The proposed method performs imitation learning by adding the target direction to the dataset in addition to the system output and camera images, which are the teacher signals, while moving with an autonomous system that uses a range-finding sensor during learning. After learning, the system selects a route based on the camera images and the target direction for autonomous moving. However, research has shown that this learning process requires a lot of time. Therefore, two approaches are proposed to reduce the learning time. In order to verify the effectiveness of these proposals, we conducted experiments using a simulator and in a real environment.

keywords: End-to-end learning, Navigation, Target direction

# 目次

第 1 章	序論	1
1.1	背景	1
1.2	目的	7
1.3	論文構成	7
第 2 章	要素技術	8
2.1	Deep learning	8
2.2	end-to-end 学習	9
2.3	Convolution Neural Network	10
2.4	地図を用いたルールベースの制御器	12
2.5	従来手法	13
2.5.1	学習フェーズ	14
2.5.2	テストフェーズ	15
第 3 章	提案手法	16
3.1	提案手法の概要	16
3.2	学習フェーズ	18
3.3	テストフェーズ	19
3.4	目標方向	20
3.5	ネットワーク構造	21
第 4 章	提案手法の検証実験	22
4.1	実験目的	22

4.2	実験装置	22
4.2.1	コンピュータとシミュレータ	22
4.2.2	ロボット	23
4.2.3	環境	23
4.3	実験方法	24
4.4	実験結果	26
<b>第5章 学習時間の短縮化の提案</b>		29
5.1	問題	29
5.2	学習step数の削減の影響	30
5.3	アプローチ1：データセットに加えるデータの割合の変更	31
5.3.1	実験目的	31
5.3.2	実験装置	32
5.3.3	実験方法	32
5.3.4	実験結果	32
5.4	アプローチ2：学習フェーズにおける積極的な蛇行	33
5.4.1	実験目的	34
5.4.2	実験装置	34
5.4.3	実験方法	34
5.4.4	実験結果	35
5.5	実環境に似たシミュレータ環境による実験	39
5.6	実環境の実験	41
5.6.1	実験目的	41
5.6.2	実験装置	41
5.6.3	実験方法	42
5.6.4	実験結果	43
<b>第6章 結論</b>		45
<b>参考文献</b>		46

目次

vii

付録	48
----	----

謝辞	49
----	----

# 図目次

1.1	Training the neural network from [1] . . . . .	1
1.2	The trained network is used generate steering commands from a single front-facing center camera from [1] . . . . .	2
1.3	How the CNN “ sees ” an unpaved road. Top: subset of the camera image sent to the CNN. Bottom left: Activation of the first layer feature maps. Bottom right: Activation of the second layer feature maps. from [1] . . . . .	2
1.4	Structure of the proposed system . . . . .	3
1.5	A robot that follows a path using vision based on the proposed method from [2] . . . . .	3
1.6	A fork in the road where the direction of travel is not unique . . . . .	4
1.7	Two network architectures for command-conditional imitation learning from [3] . . . . .	5
1.8	End-to-end driving via conditional imitation learning from [3] . . . . .	5
1.9	Model structure from [4] . . . . .	6
2.1	Neural network . . . . .	8
2.2	Structure of end-to-end learning . . . . .	9
2.3	AlexNet from [5] . . . . .	10
2.4	VGG from [5] . . . . .	11
2.5	A rule-based controller using a map . . . . .	12
2.6	Learning phase of conventional method from [6] . . . . .	13
2.7	Output of rule-based controller using a map and actual robot behavior . . . . .	14

2.8	Test phase of conventional method . . . . .	15
3.1	Overall flow of the proposed method . . . . .	16
3.2	Topological map . . . . .	17
3.3	Learning phase system of conventional method . . . . .	18
3.4	Learning phase system of proposed method . . . . .	18
3.5	Test phase system of proposed method . . . . .	19
3.6	Target direction . . . . .	20
3.7	Structure of network . . . . .	21
4.1	TurtleBot3 waffle_pi with 3 cameras . . . . .	23
4.2	Experimental environment of simulator . . . . .	23
4.3	Characteristics of passages in the experimental environment on the simulator	24
4.4	Moving pattern at points A and B . . . . .	24
4.5	The order of the route to be moved . . . . .	25
4.6	Experimental results for each moving pattern from [6] . . . . .	26
4.7	Foyer on the route to be moved . . . . .	27
4.8	Experimental results for each moving pattern at 6000step by real environment . . . . .	28
5.1	Experimental results for each moving pattern at 10000step . . . . .	30
5.2	Number of data per command per 10000steps in conventional experiments	31
5.3	Experimental results for each moving pattern by CIL with CoDP . . . . .	32
5.4	Moving on the target path . . . . .	33
5.5	Robot behavior . . . . .	34
5.6	Moving on the target path attempting CIL with CoDP+AM . . . . .	34
5.7	Experimental results for each moving pattern at 10000step by CIL with CoDP+AM . . . . .	35
5.8	Ratio of data by distance from target path in learning phase . . . . .	37
5.9	Ratio of data by distance from target path in test phase . . . . .	37

5.10	Experimental results for each moving pattern at 20000step by CIL with CoDP+AM . . . . .	38
5.11	Simple simulator environment . . . . .	39
5.12	Simulator environment similar to real environment . . . . .	39
5.13	Experimental results for each moving pattern at 20000step by Approach 1+2(simulator environment to real environment) . . . . .	40
5.14	Experimental setup . . . . .	41
5.15	Real environment . . . . .	42
5.16	Experimental results for each moving pattern by real environment . . . . .	43
5.17	Acquired camera images at corresponding places . . . . .	44
5.18	Experimental results for each moving pattern by models with the highest success rate . . . . .	44

# 表目次

2.1	Angular velocity offset . . . . .	14
3.1	Target direction list . . . . .	20
3.2	Parameters of network . . . . .	21
4.1	Experimental results . . . . .	26
4.2	Experimental results . . . . .	27
5.1	Experimental results at 10000step . . . . .	30
5.2	Experimental results by CIL with CoDP . . . . .	33
5.3	Experimental results at 10000step by CIL with CoDP+AM . . . . .	35
5.4	Experimental results at 20000step by CIL with CoDP+AM . . . . .	38
5.5	Experimental results by simulator similar to real environment . . . . .	40
5.6	Experimental results by real environment . . . . .	43

# 第1章

## 序論

### 1.1 背景

近年、機械学習を用いた自律走行に関する研究が盛んにされており、その中でカメラ画像を用いる研究もされている。Bojarski ら [1] は Fig. 1.1 に示すシステムで、人間のドライバーが操作するステアリング角度と前方カメラ画像を用いて模倣学習を行った。加えて、Fig. 1.2 に示すように、訓練したネットワークに画像を入力し、生成される操舵指令を用いて走行を行う手法を提案した。

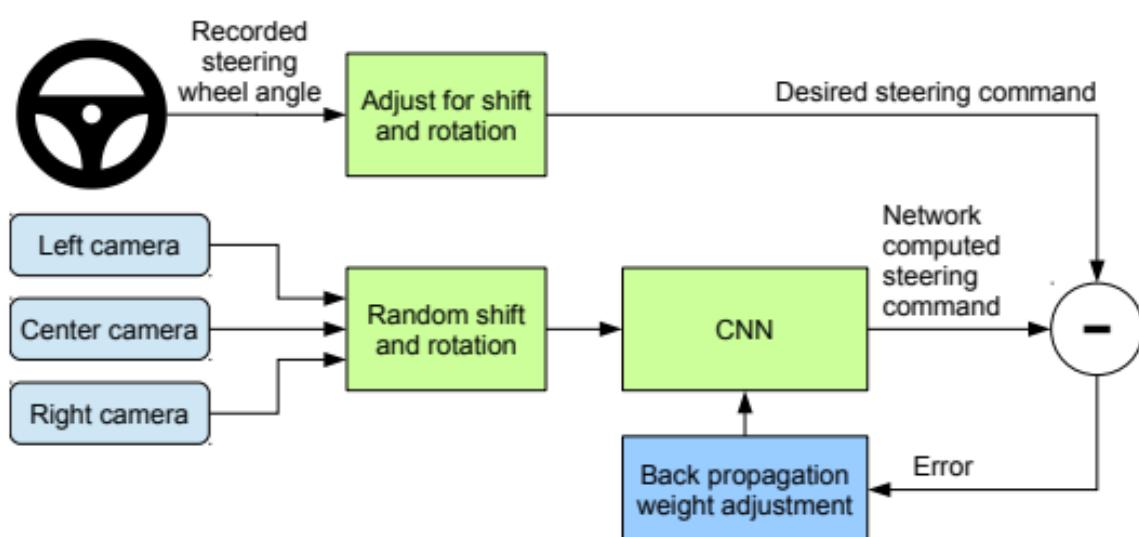
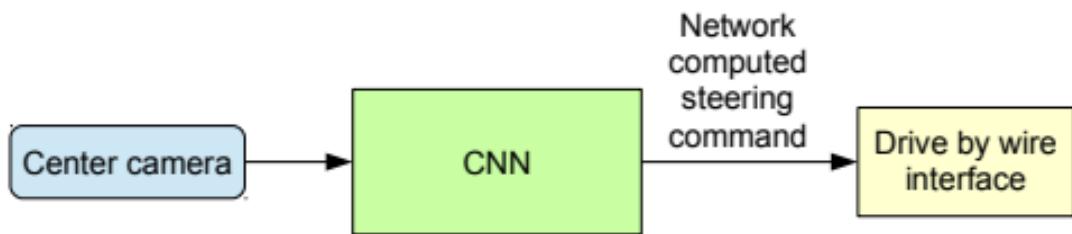
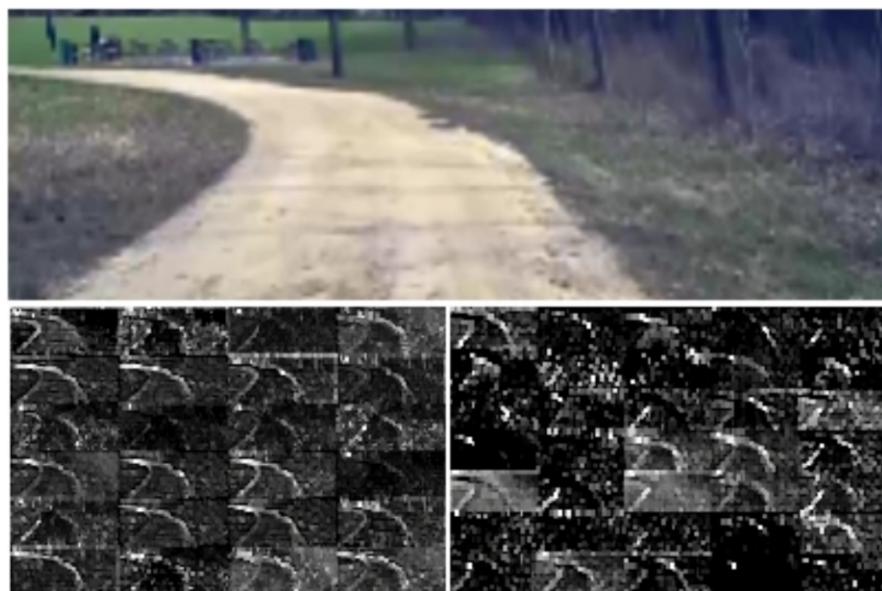


Fig. 1.1: Training the neural network from [1]



**Fig. 1.2:** The trained network is used generate steering commands from a single front-facing center camera from [1]

また, Fig. 1.3 は CNN が人間のドライバーが操作するステアリング角度を教師信号として, 単独で有用な道路の特徴を検出するように学習したことを示している. なお, 道路の輪郭を検出するように明示的に学習をさせていない.



**Fig. 1.3:** How the CNN “sees” an unpaved road. Top: subset of the camera image sent to the CNN. Bottom left: Activation of the first layer feature maps. Bottom right: Activation of the second layer feature maps. from [1]

本研究室においても、岡田ら [2][7] は Fig. 1.4 に示すようなシステムを用いて Fig. 1.5 のように経路追従行動を模倣学習し、カメラ画像に基づいた経路追従行動を獲得した。このシステムでは、LiDAR、オドメトリを入力としたルールベース制御器（後述する”地図を用いたルールベース制御器”）による経路追従行動を前方カメラ画像を用いて end-to-end で模倣学習した。

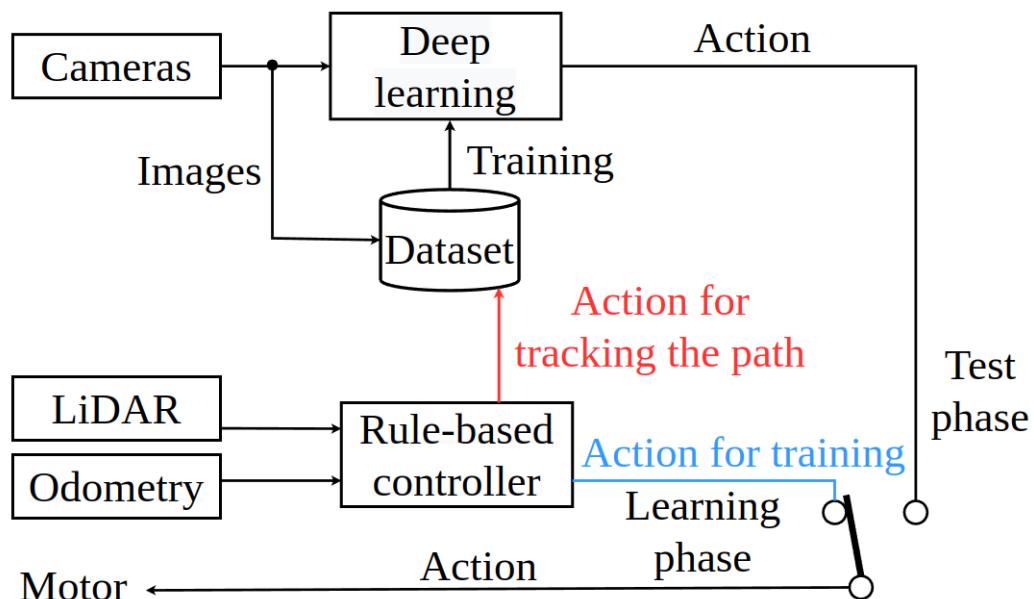


Fig. 1.4: Structure of the proposed system



Fig. 1.5: A robot that follows a path using vision based on the proposed method from [2]

上記の研究により、カメラ画像に基づいてロボットが学習した経路を周回可能であることが確認されている。しかし、岡田らの手法は学習を行った一定の経路を走行できるが、目的地に対して経路を動的に選択して走行することはできない。それをするためには、以下のような拡張が必要となる。

- ネットワークの入力に新たな要素の追加

そのため、岡田らの研究（以下、「従来手法」と称する）をベースに、ネットワークへ新たな要素を追加し、Fig. 1.6 のような分岐路において、任意の経路を選択する機能の追加を検討する。



Fig. 1.6: A fork in the road where the direction of travel is not unique

カメラ画像とステアリング角度に、条件を加えて学習を行う条件付き模倣学習の研究は、現在までにいくつか行われている。Felipe ら [3] は前方カメラ画像、ステアリング角度、加速度と「continue」、「left」、「straight」、「right」からなるコマンドを入力とした Fig. 1.7 のようなネットワークを用いて、Fig. 1.8 に示すような実環境と都市環境のシミュレータ上で、模倣学習のテスト時においてもコマンドによって制御可能であることを確認している。

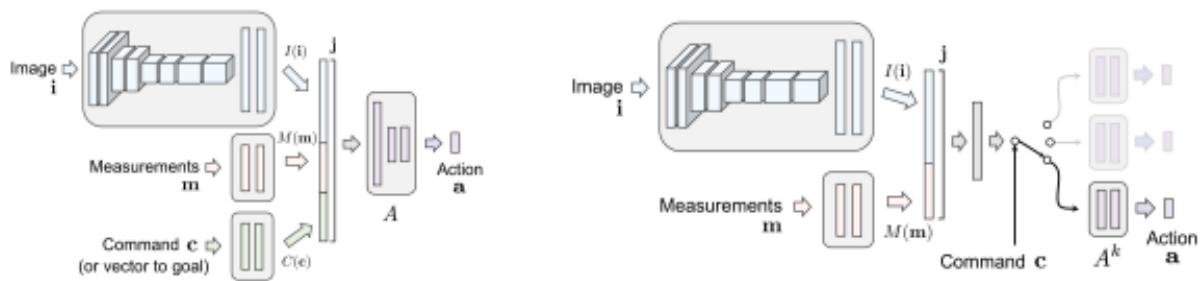


Fig. 1.7: Two network architectures for command-conditional imitation learning from [3]

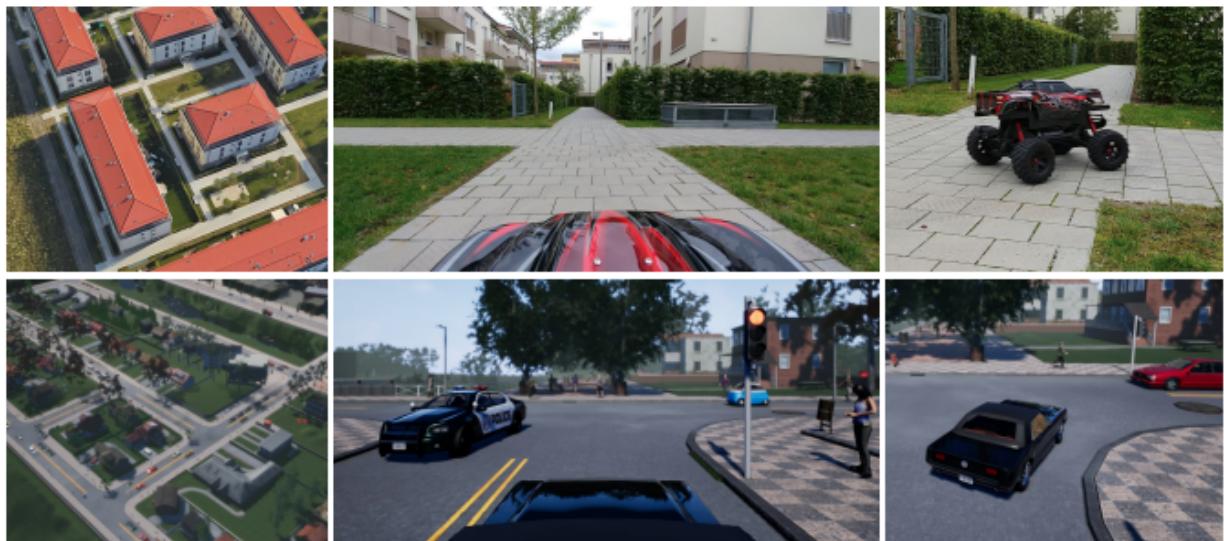


Fig. 1.8: End-to-end driving via conditional imitation learning from [3]

また, Hawke ら [4] は Fig. 1.9 のような, 3 つの前方カメラ画像と「go-straight」, 「turn-left」, 「turn-right」からなるコマンドを入力とする構造のモデルを用いて, 実環境での複雑な都市環境というシナリオで, 意思決定が可能なモデルをわずか 30 時間の学習データで学習可能であることを示している. ただし, これらの研究は, 本研究の提案するオンラインで自動的にデータセットを収集して学習する仕組みを持っておらず, その点で本研究室で提案している手法とは異なっているといえる.

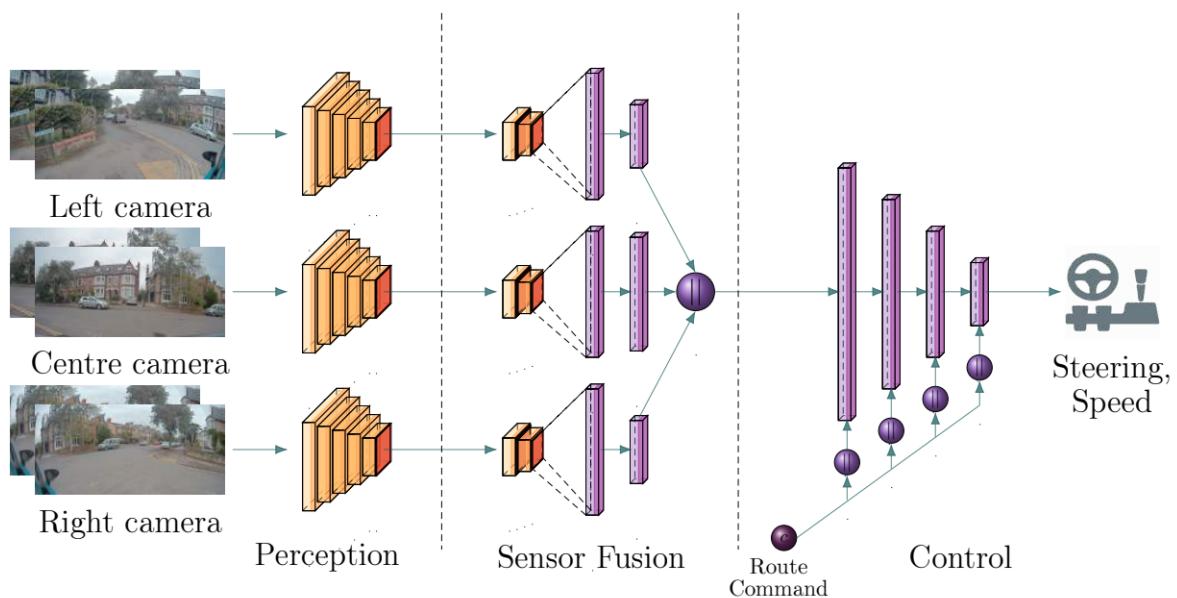


Fig. 1.9: Model structure from [4]

## 1.2 目的

本研究では、経路追従行動を模倣学習し、カメラ画像に基づいた経路追従行動を獲得する手法である岡田らの従来手法をベースとして、分岐路において「直進」、「左折」などのコマンドによる制御で任意の経路を選択可能にする機能の追加を提案する。また、シミュレータ上での実験を実環境に移す際に、問題となった学習時間の長さについて、2つのアプローチの提案と検証を行う。さらに、実環境における提案手法の有効性を検証することを目的とする。

## 1.3 論文構成

1章では、本研究における背景、及び目的を述べた。2章では、本研究で用いた深層学習の要素技術とベースとする従来手法について述べる。3章では、従来手法をベースにした提案手法を述べる。4章では、シミュレータと実環境での実験を行う。5章では、本研究の結論を述べる。

## 第2章

# 要素技術

本章では、本研究で用いた深層学習に関連した要素技術と、ベースとなる従来手法について述べる。

### 2.1 Deep learning

Deep learning は、画像や音声などのデータに特に適しており、近年では自然言語処理などさまざまな分野で活用されている。人間の脳のような深い層の構造を持つ人工ニューラルネットワークに基づく機械学習手法である。人工ニューラルネットワークは、入力データから出力データを予測するために、多数のニューロンを用いて情報を処理する。この人工ニューラルネットワークを多層構造にすることで、より深い情報処理を行うことができる。これにより、高度な識別や分類タスクなどを行うことを可能にしている。一般的な構造を Fig. 2.1 に示す。

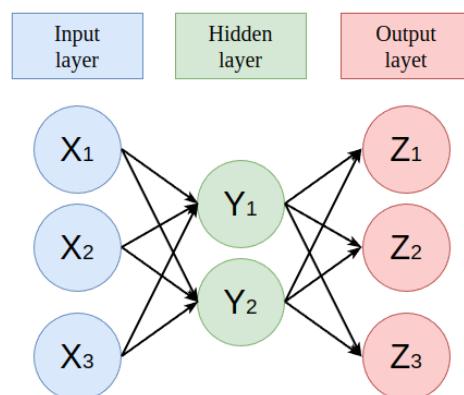


Fig. 2.1: Neural network

## 2.2 end-to-end 学習

end-to-end 学習とは, Fig. 2.2 に示すように人工ニューラルネットワークを使用して, 入力データから出力を直接生成する方法のことを指す.

実世界における自動運転を例に挙げる. end-to-end 学習を用いない場合, 人物や障害物などの物体認識, 車線の検出, 経路計画, ステアリングの制御など, 多くのタスクを解決する必要がある. しかし, end-to-end を用いることで, 先程のタスクを解決することなく, 車両が撮影したカメラ映像から直接, 運転操作を行うことができる.

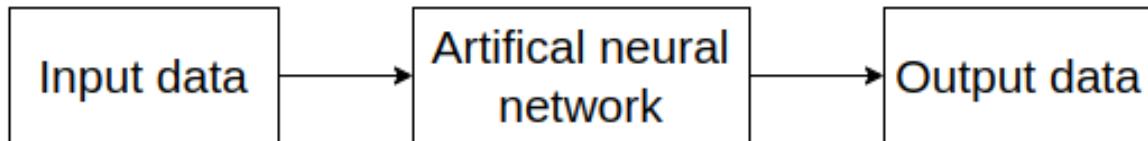


Fig. 2.2: Structure of end-to-end learning

## 2.3 Convolution Neural Network

畳み込みニューラルネットワーク (convolutional neural network:CNN) は人工ニューラルネットワークのモデルの一種である。このモデルは、画像や音声などの多次元の配列で表される複雑なデータを処理するために特別に設計されている。CNN は次のような特徴を持つ層で構成されている。

### 1. 畳み込み層

入力データをフィルタ（カーネル）を用いて特徴を抽出する。

### 2. プーリング層

特徴を残しつつ、畳み込み層の出力を圧縮する。これにより、画像であればピクセル数が減少し、計算量を大幅に減らすことができる。

### 3. 全結合層

畳み込み層とプーリング層の出力をまとめて処理する。

Krizhevsky ら [5] は Fig. 2.3 で示すような、8 層のネットワークを用いて、画像分類タスクをエラー率 15.3% で達成し、画像分類コンペティションである ILSVRC(ImageNet Large Scale VisualRecognition Competition)2012 で優勝した。

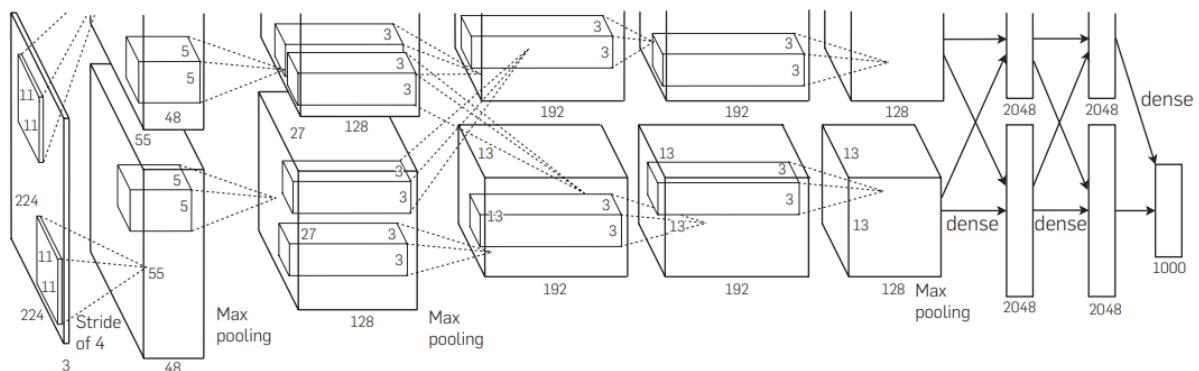


Fig. 2.3: AlexNet from [5]

Simonyan[8] らは CNN の層の深さが精度に与える影響を調査した。Fig. 2.4 のような、最大 19 層の深い畳み込みネットワークを評価した。その結果、モデルを深層にすることが分類精度に有利であることが示された。ILSVRC2012 の優勝モデルである AlexNet は 8 層、ILSVRC2013 で提案された ZFNet は同様の 8 層であることから、当時の CNN としては圧倒的に深い層を持つモデルであった。このような深い畳み込みネットワークは、深層学習における重要な発展の一つとされている。

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input ( $224 \times 224$ RGB image)					
conv3-64	conv3-64 LRN	conv3-64 <b>conv3-64</b>	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 <b>conv3-128</b>	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 <b>conv1-256</b>	conv3-256 conv3-256 <b>conv3-256</b>	conv3-256 conv3-256 <b>conv3-256</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Fig. 2.4: VGG from [5]

## 2.4 地図を用いたルールベースの制御器

後述する従来手法と提案手法において、教師信号として用いる地図を用いたルールベース制御器について述べる。地図を用いたルールベース制御器は、ROS Navigation\_stack[9] へ目標位置 ( waypoint ) の指示を行う waypoint\_nav[10] を組み合わせたものである。なお、後述するが提案手法では waypoint\_nav の役割が増えている。ROS Navigation\_stack では以下のようないくつかの処理が行われる。

- ロボットの現在位置を推定する
- 移動目標地点までの経路を決定する
- 経路に追従する行動をロボットに指示する

また、Fig. 2.5 に示すように、事前に作成した占有格子地図と測域センサ、オドメトリを用いて、地図上での自己位置を particle filter を用いて近似することで推定する「amcl」。障害物認識などによる局所的、または地図全体の大域的なコスト計算、その結果に基づいた経路計画、それに従ったモータ指令を行う「move\_base」などのパッケージによって構成されている。従来手法、提案手法ともにモータ指令を並進速度は一定とし、角速度のみを制御対象とした。

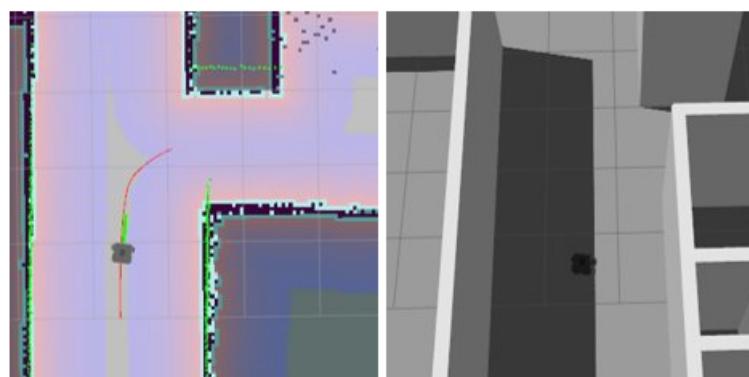


Fig. 2.5: A rule-based controller using a map

## 2.5 従来手法

本研究のベースとなる岡田らの研究について述べる。先に述べたように、本論文では岡田らの手法を「従来手法」と呼ぶ。従来手法は、地図を用いたルールベース制御器による走行を模倣学習し、似た行動を画像を用いて行う手法である。

Fig. 2.6 に、経路追従行動を視覚に基づいてオンラインで模倣するシステムを示す。手法は模倣学習により、学習器の訓練をする学習フェーズと訓練した結果を検証するテストフェーズに分かれる。なお、両フェーズで用いる並進速度は一定の値を用いる。

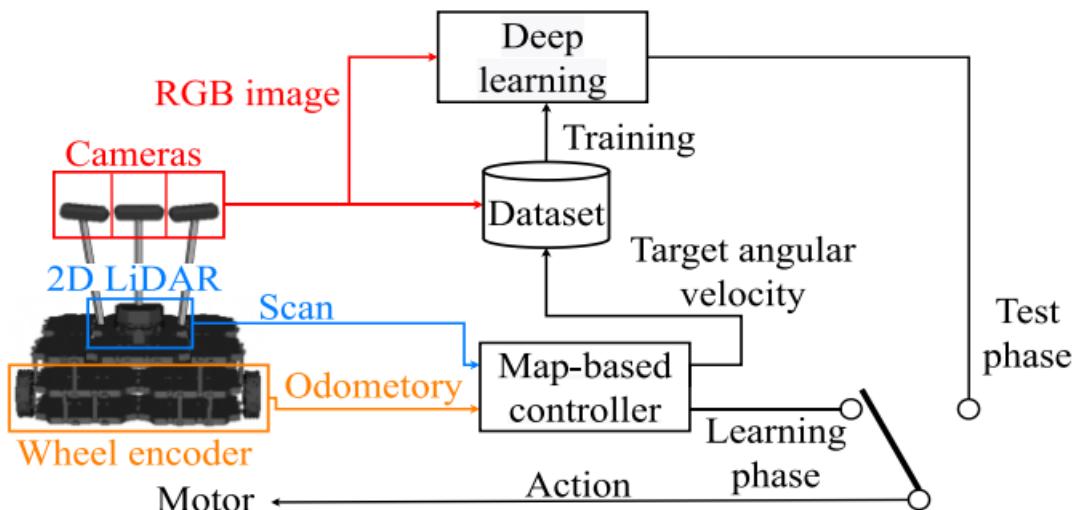


Fig. 2.6: Learning phase of conventional method from [6]

### 2.5.1 学習フェーズ

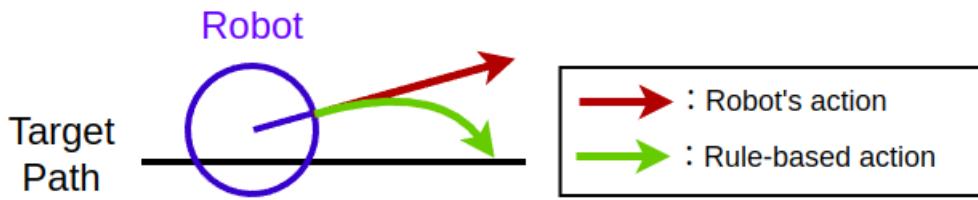
学習フェーズは、模倣学習によって学習器の訓練を行うフェーズである。測域センサとオドメトリを入力とする地図を用いたルールベース制御器で自律移動する。具体的には、ROS navigation\_stack パッケージを利用して、ロボットに自律移動させる。学習フェーズでは、ロボットの中央、左、右に傾けて取り付けた3つのカメラを用いて画像を取得する。

自律移動させる際に、取得するデータ量を増加させること、及び過学習の抑制を目的として、Table 2.1 に示すような処理を行う。また、地図を用いたルールベース制御器による走行をそのまま模倣学習するのではなく、少し蛇行するように自律移動させることで、経路に戻るような挙動も学習できるようになっている。Fig. 2.7 に示すように、実際にロボットを制御する行動と経路に従う行動を別に扱うことで、常に経路に従う行動をデータセットに加えることを可能にしている。

岡田らの手法では、データセットの収集方法にいくつかの種類がある。本論文では、その中最も経路追従の成功率が高い手法を用いて、ロボットに模倣学習をさせる。

**Table 2.1:** Angular velocity offset

Left camera	Angular velocity of a rule-based controller using a map + 0.2 rad/s
Center camera	Angular velocity of a rule-based controller using a map
Right camera	Angular velocity of a rule-based controller using a map - 0.2 rad/s



**Fig. 2.7:** Output of rule-based controller using a map and actual robot behavior

### 2.5.2 テストフェーズ

学習器の訓練後, Fig. 2.8 で示すテストフェーズへ移行する. このフェーズでは, 学習器にカメラ画像を入力し, 出力されるヨー方向の角速度を用いて自律移動することで, 訓練後の学習結果を評価する. なお, テストフェーズでは中央のカメラのみを用いる.

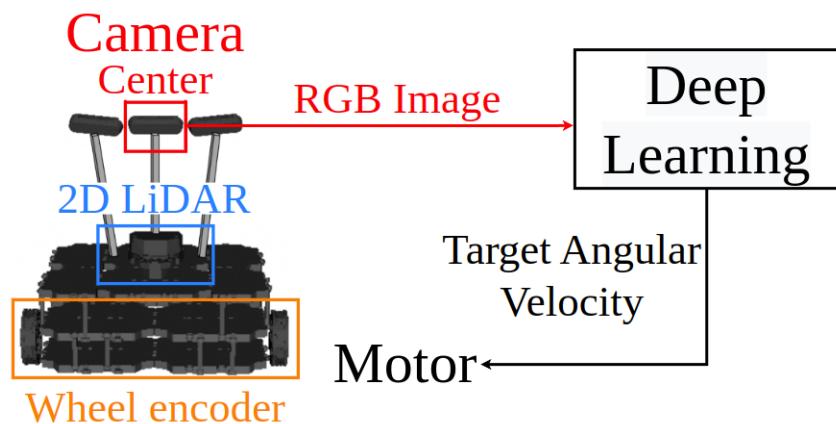


Fig. 2.8: Test phase of conventional method

## 第3章

# 提案手法

本章では、従来手法をベースとする提案手法についての概要、提案手法における学習フェーズ、テストフェーズ、目標方向、ネットワーク構造についての5節に分けて述べる。

### 3.1 提案手法の概要

本研究では、従来手法をベースに「直進」、「左折」、「右折」の目標とする進行方向の情報（以下、「目標方向」と称する）をデータセットと学習器へ追加する。これにより、訓練済みの学習器の出力を用いた走行において、目標方向により任意の経路を選択可能とする機能の追加を行った。なお、追加した要素以外は従来手法と同様である。提案手法全体の流れを Fig. 3.1 に示す。

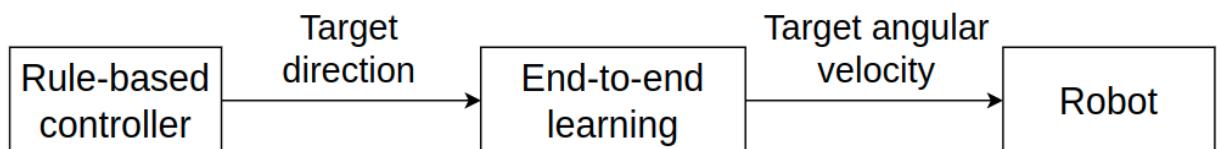


Fig. 3.1: Overall flow of the proposed method

最終的にはカメラ画像を入力として、トポロジカルマップによって生成される目標方向に従って、目的地まで移動する自律走行の手法を提案することを検討する。トポロジカルマップとは Fig. 3.2 に示すように、分岐路などの目印（ノード）とつながり（エッジ）を持つ簡略化された地図である。

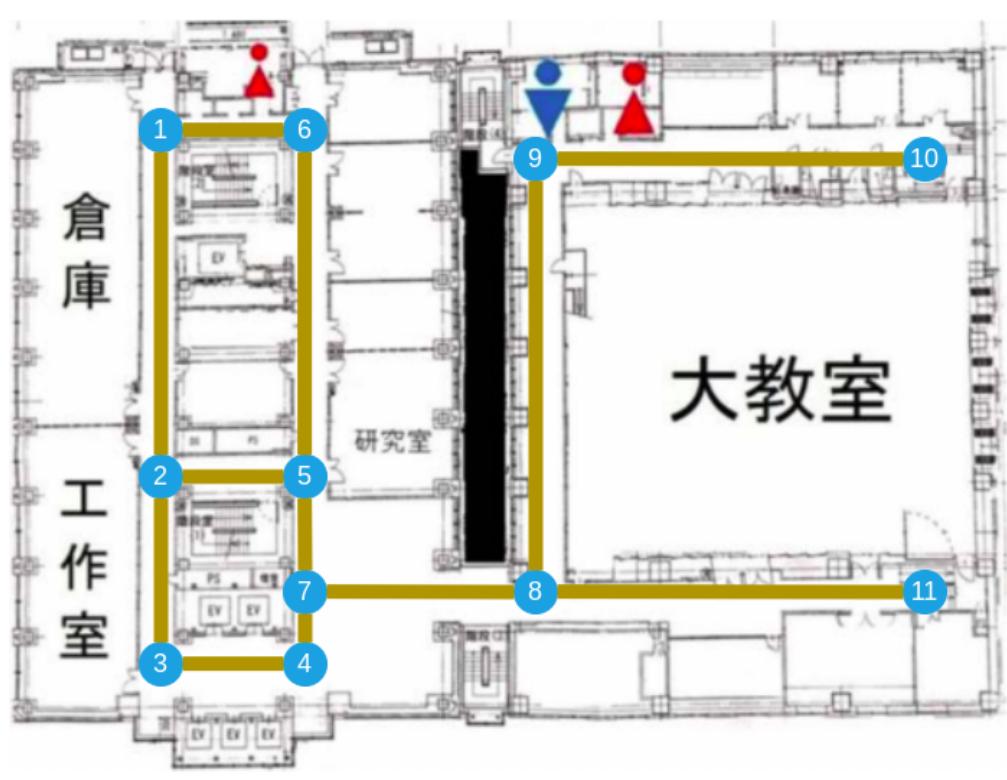


Fig. 3.2: Topological map

### 3.2 学習フェーズ

提案手法で用いる学習フェーズのシステムを Fig. 3.4 に示す。自律移動を行う地図を用いたルールベース制御器から目標方向を生成し、データセットに加えている。なお、厳密にはルールベース制御を構成する waypoint\_nav により目標方向を生成している。提案手法では、LiDAR とオドメトリを入力とする地図を用いたルールベース制御器による自律移動を、カメラ画像と目標方向を用いて模倣学習する。

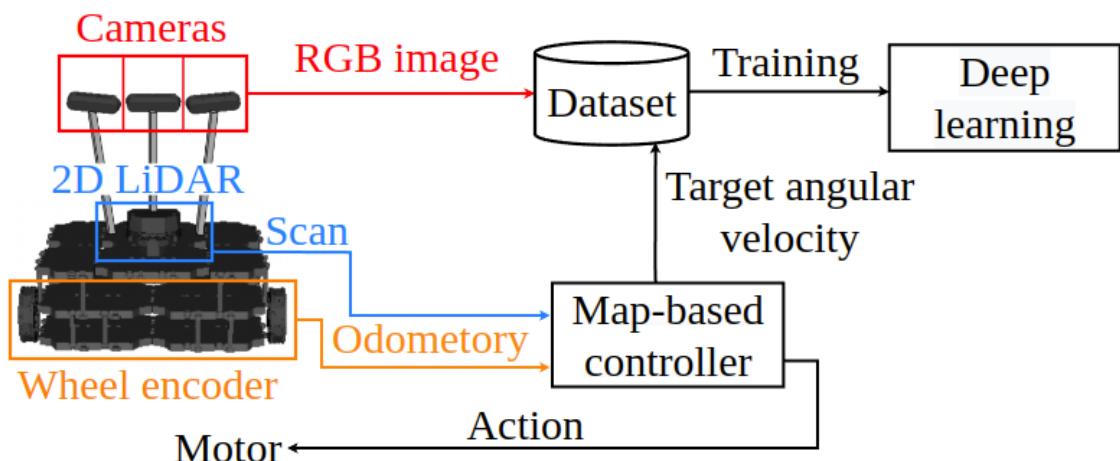


Fig. 3.3: Learning phase system of conventional method

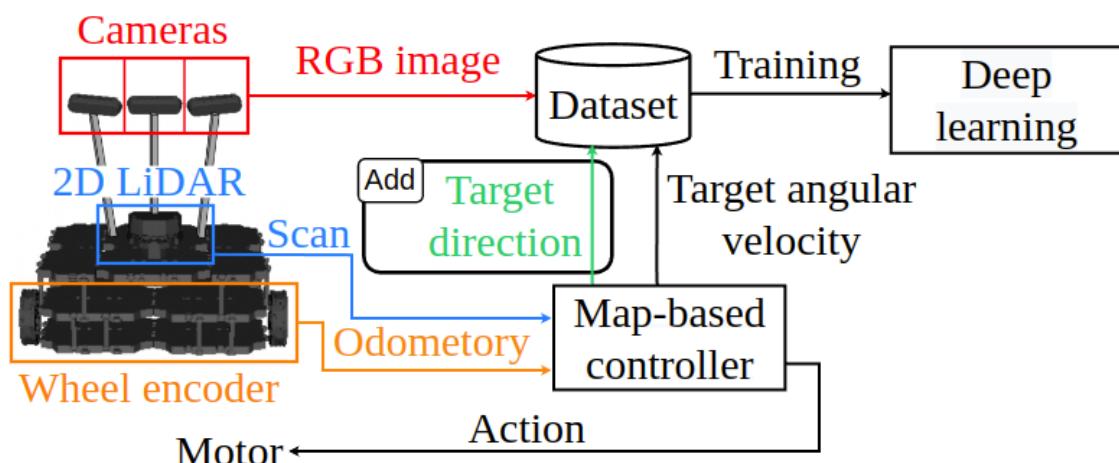


Fig. 3.4: Learning phase system of proposed method

### 3.3 テストフェーズ

提案手法におけるテストフェーズでは Fig. 3.5 で示すように、従来手法のシステムから新たに目標方向を学習器の入力へ追加した。なお、テストフェーズも学習フェーズと同様に、地図を用いたルールベース制御器から目標方向を生成している。

最終的には、目的地までカメラ画像のみで自律移動させることを目標としているため、目標方向を画像から自動的に作成する仕組みが必要となるが、それは今後の課題として、本論文では議論しない。

学習フェーズで訓練した学習器の出力により自律走行しながら、目標方向によって任意の経路を選択する。

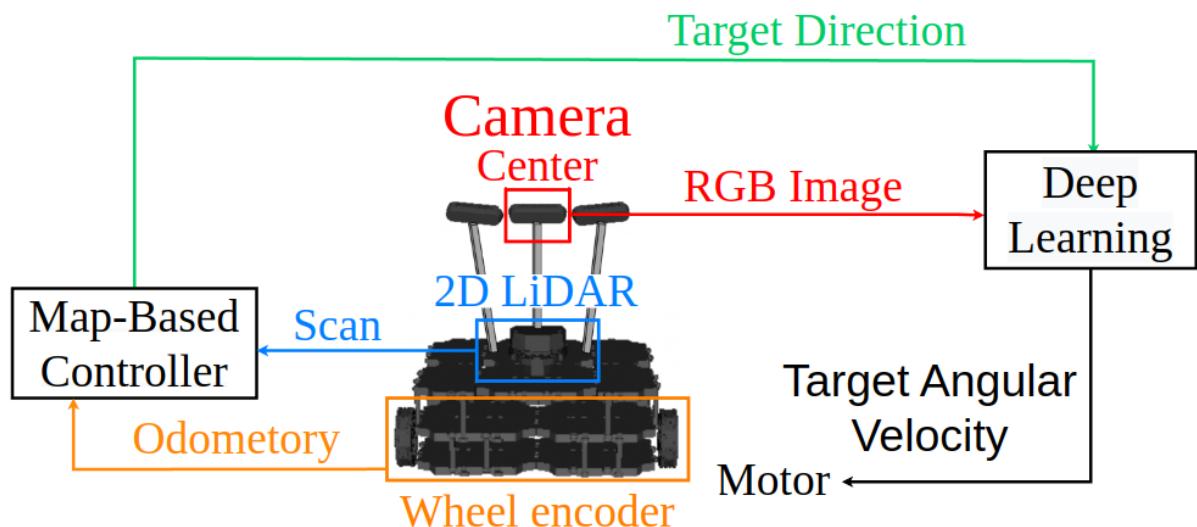
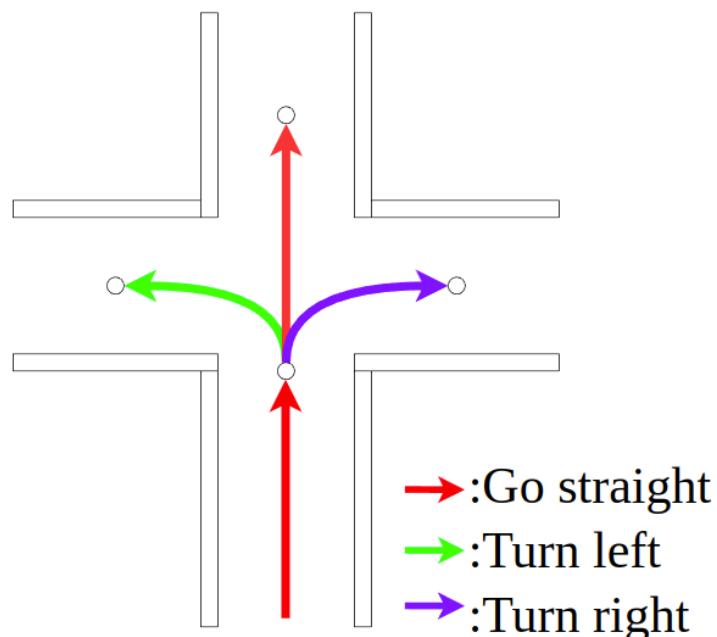


Fig. 3.5: Test phase system of proposed method

### 3.4 目標方向

本研究で用いた目標方向と、そのデータ形式である目標方向指令について述べる。目標方向を Fig. 3.6 に示す。目標方向を経路と分岐路において「道なり」に走行 (Go straight)，分岐路において「直進 (Go straight)」，「左折 (Turn left)」，「右折 (Turn right)」の 3 つとする。なお、文献 [6] では「道なり (continue)」，「直進 (Go straight)」，「左折 (Turn left)」，「右折 (Turn right)」からなる 4 つのコマンドを用いていたが、「道なり」と「直進」のコマンドに対応する行動がほぼ同一であったため、本研究ではコマンドを 1 つ減らし、3 コマンドとしている。



**Fig. 3.6:** Target direction

学習器には、上記の 3 つの目標方向を要素数 3、次元数 1 の int 型の配列で表現した “目標方向指令” を入力する。目標方向指令のデータ形式を Table 3.1 に示す。

**Table 3.1:** Target direction list

Target Direction	Go straight	Turn left	Trun right
Data	[100, 0, 0]	[0, 100, 0]	[0, 0, 100]

### 3.5 ネットワーク構造

提案手法で用いた学習器のネットワークを Fig. 3.7 に示す。また、ハイパーパラメータについて Table 3.2 に示す。64x48 の RGB 画像を入力とする入力層 1 層、畳み込み層 3 層、全結合層 2 層を持つ 6 層の CNN と、この CNN の出力と目標方向指令を入力する入力層 1 層、全結合層 2 層、出力層 1 層の全 10 層から構成されている。出力はヨー方向の角速度である。

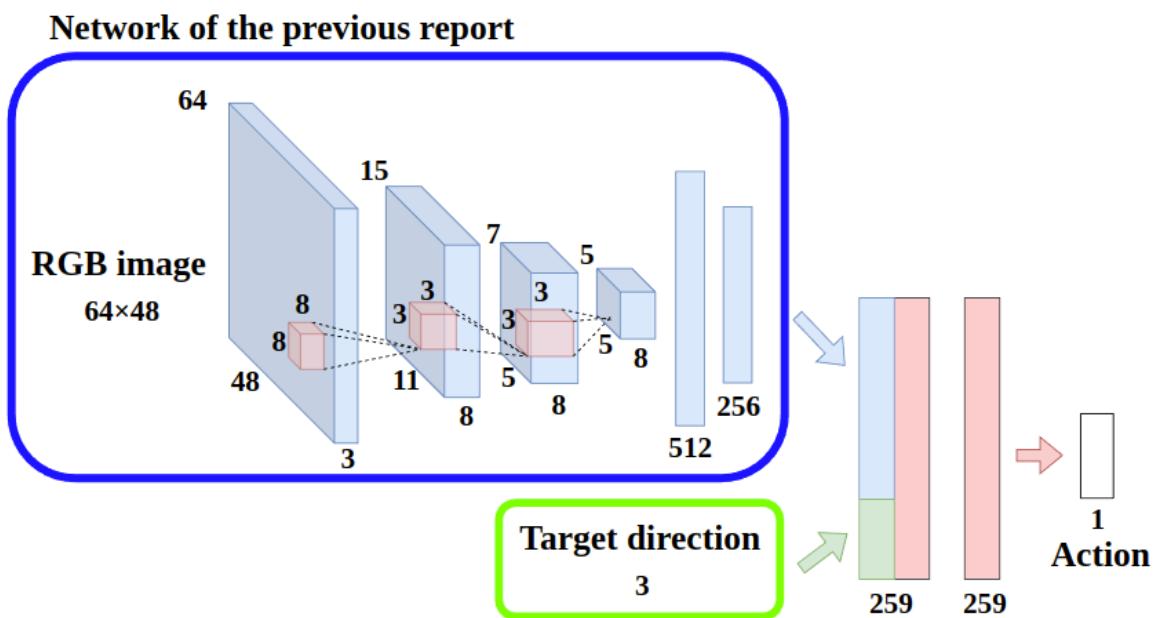


Fig. 3.7: Structure of network

Table 3.2: Parameters of network

Input data	Image(64x48 pixels, RGB channels), Target direction
Optimizer	Adam( $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e^{-2}$ )
Loss function	Softmax-cross-entropy
Output data	Angular velocity

## 第4章

# 提案手法の検証実験

### 4.1 実験目的

シミュレータ上と実環境で、提案手法の有効性の検証を行う

### 4.2 実験装置

#### 4.2.1 コンピュータとシミュレータ

##### 1. コンピュータ

OS: Ubuntu 20.04 LTS

ROS: Noetic

CPU: intel Core i7-10700F(4.8GHz/8コア/16スレッド)

DRAM: 32GB DDR4(3200/8GB × 4)

##### 2. nav\_cloning(学習器、統合環境)

[https://github.com/open-rdc/nav\\_cloning](https://github.com/open-rdc/nav_cloning)

##### 3. waypoint\_nav(移動目標地点、目標方向を出力)

[https://github.com/open-rdc/waypoint\\_nav](https://github.com/open-rdc/waypoint_nav)

##### 4. turtlebot3 関連

<https://github.com/open-rdc/turtlebot3>

### 5. navigation(ナビゲーションパッケージ)

<https://github.com/ros-planning/navigation>

#### 4.2.2 ロボット

ロボットモデルは従来手法の検証に用いた実験 [2][7] と同様, Fig. 4.1 に示すような, TurtleBot3 Waffle\_pi[11] へ 3 つのカメラを追加したモデルを用いる.



Fig. 4.1: TurtleBot3 waffle\_pi with 3 cameras

#### 4.2.3 環境

シミュレータ環境として, オープンソースの 3D ロボットシミュレータ Gazebo[12] を用いる. Fig. 4.2 に示すような Gazebo 上で千葉工業大学津田沼キャンパス 2 号館 3 階を模した実験環境を対象に実験を行う.

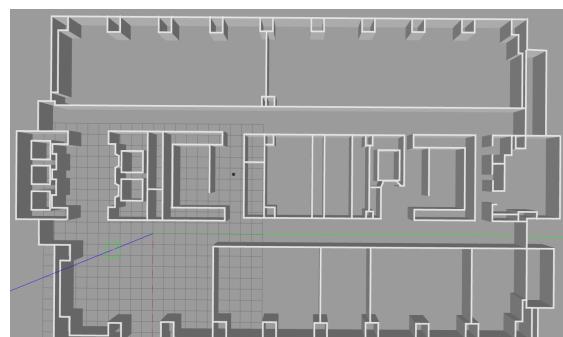


Fig. 4.2: Experimental environment of simulator

### 4.3 実験方法

Fig. 4.3 の A, B 地点において, Fig. 4.4 に示すように侵入する方向が 3 つあり, 進むことのできる方向が 2 つあることから, 1 箇所につき走行パターンが 6 つ存在する. また, A, B 地点では, 目標方向に従って任意の経路を選択することが求められる場所である. したがって, 実験では A, B 地点で合計 12 パターンの走行において, 与えた目標方向に従った行動が行えるのかを確認する.

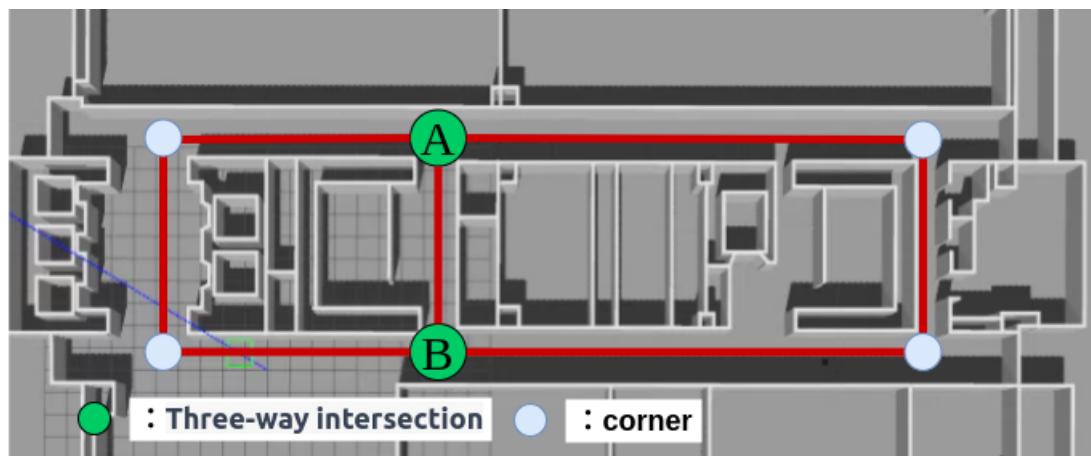


Fig. 4.3: Characteristics of passages in the experimental environment on the simulator

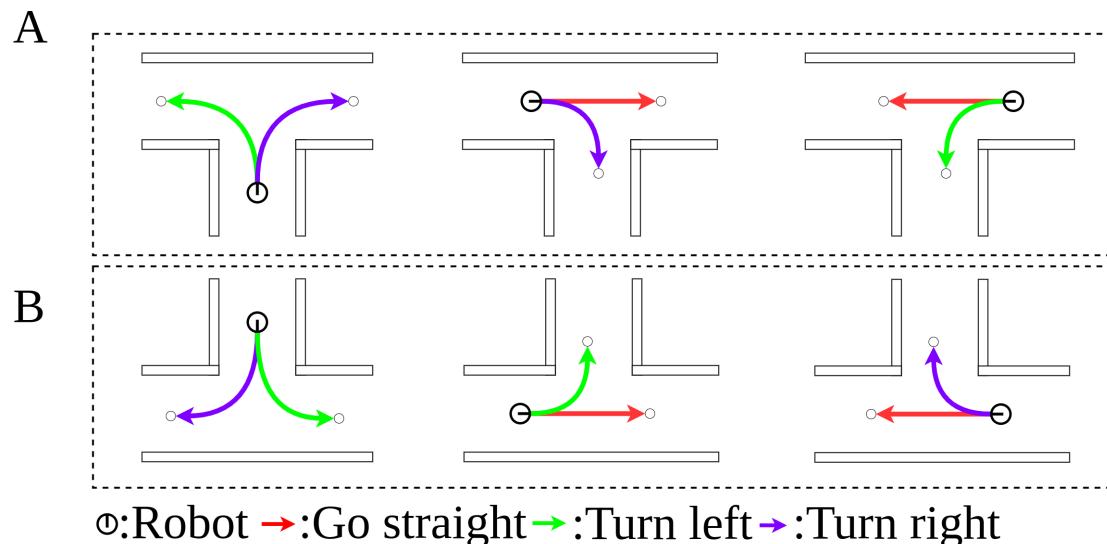


Fig. 4.4: Moving pattern at points A and B

全ての走行パターンを網羅するように模倣学習を行うため, Fig. 4.5 に示すように a から f までの経路を繰り返し走行させる。なお、目標方向は waypoint\_nav から生成され、データセットに加えられる。学習終了後、テストフェーズに移行するが、学習時と同様に a から f までの順番で経路をロボットに走行させる。また、テストフェーズ時にロボットが壁に衝突した場合、経路の中央にロボットを移動させた後、走行を再開させる。

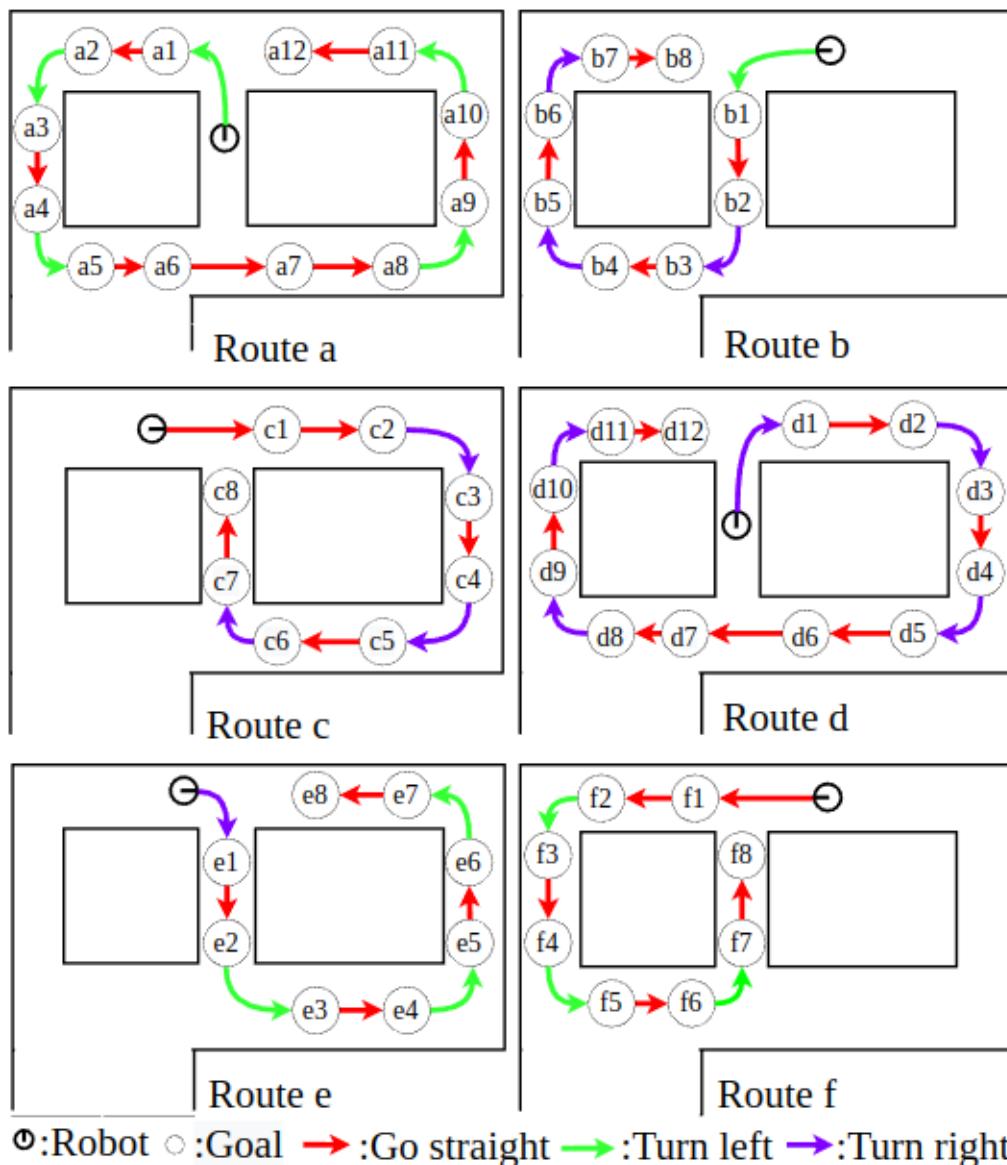


Fig. 4.5: The order of the route to be moved

学習を 60000step 実行後、テストフェーズに移行する。テストフェーズで正しい順序で経路を選択し、走行を行えるか確認する。この一連の流れを 10 回繰り返し行う。

## 4.4 実験結果

実験結果を Fig. 4.6 に示す。この図は、それぞれの走行パターンにおいて正しく経路を選択し、走行できた回数を表している。Table 4.1 に全パターンの成功回数を合計した結果を示す。なお、分母が 120 であるのはテストフェーズにおいて、全 12 パターンからなる経路を走行させ、評価を行うことを 10 回繰り返したためである。

Table 4.1 に示すように、目標方向に従って 113/120 回、正しい経路を選択する様子が見られた。成功率が約 94% となったことから、簡易的なシミュレータ上では提案手法により、経路を選択できることが判明した。しかし、Fig. 4.6 の b3, e3 に向かう選択ができていない。特に、b3 に関しては成功回数が著しく少ない。これは、Fig. 4.7 に示すように、b3 が評価対象である 12 パターンの中で、唯一ホワイエに向かう経路であり、他のパターンに比べて急激に取得するカメラ画像の特徴が変化することが影響したおそれがある。また、e3 に関しては、カーブを行い始める段階で、取得するカメラ画像にホワイエが写り込んでしまうことが影響したおそれがある。

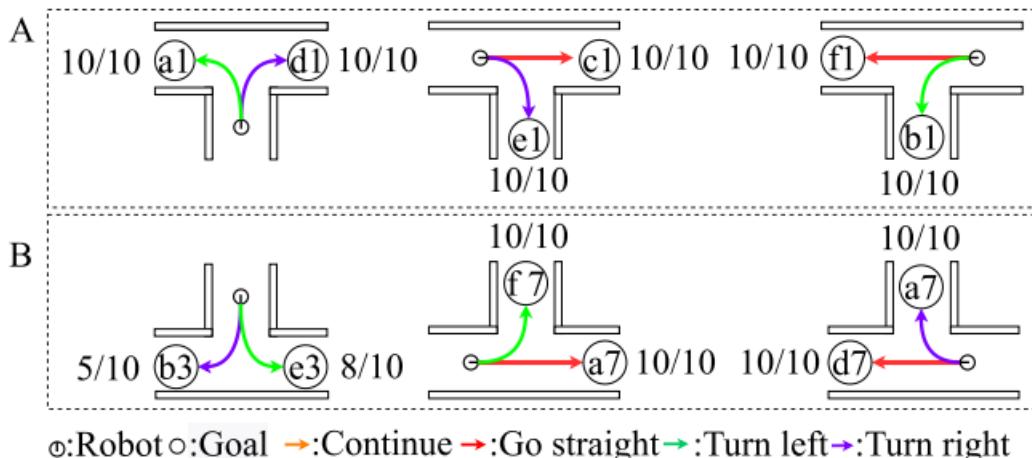


Fig. 4.6: Experimental results for each moving pattern from [6]

Table 4.1: Experimental results

Step	Total result
60000	113/120 (94.2%)

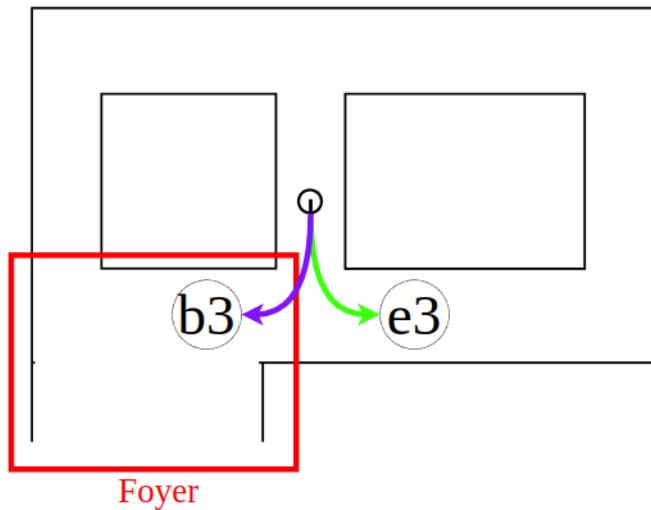


Fig. 4.7: Foyer on the route to be moved

次に、実環境での実験を行った。実験方法は、シミュレータ上での実験に倣って設定した。しかし、実験を行ったのは1回のみである。実験が1回のみなのは、学習時間の長さが問題であったためである。具体的には、実験にはおよそ7時間を要し、バッテリなどの関係で1日に1,2時間が限界であったため、1週間かけて実験を行う必要があった。なお、実験装置については後述する。

実験結果を Fig. 4.8 に示す。この図は、それぞれの走行パターンにおいて正しく経路を選択し、走行できた回数を表している。Table 4.2 に実験ごとに全パターンの成功回数を合計した結果を示す。Table 4.2 に示すように、目標方向に従って 9/12 回、正しい経路を選択する様子が見られた。

Table 4.2: Experimental results

Experiments	Step	Total result
Simulator	60000	113/120 ( 94.2% )
Real environment	60000	9/12 ( 75% )

Table 4.2 から、シミュレータ上と実環境で成功回数に大きく差があることがわかる。これについて、5.6.4 項で述べる。

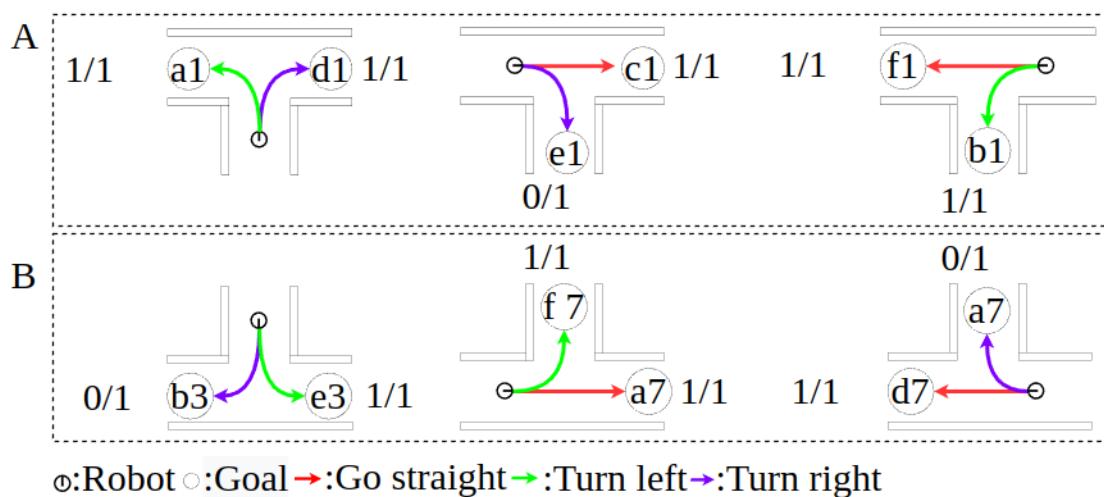


Fig. 4.8: Experimental results for each moving pattern at 60000step by real environment

## 第5章

# 学習時間の短縮化の提案

この章では、1節で、前章で問題となった学習時間の長さについて議論する。2節では、前章の実験を基に、単に学習量を減らした実験を行う。3, 4節では、問題を解決するための2つのアプローチをそれぞれ提案し、実験と検証を行う。5節では、実験に簡易的なシミュレータを用いる問題点を述べ、解決策を提示する。6節では、実環境で実験を行い、実環境における提案手法の有効性を検証する。

### 5.1 問題

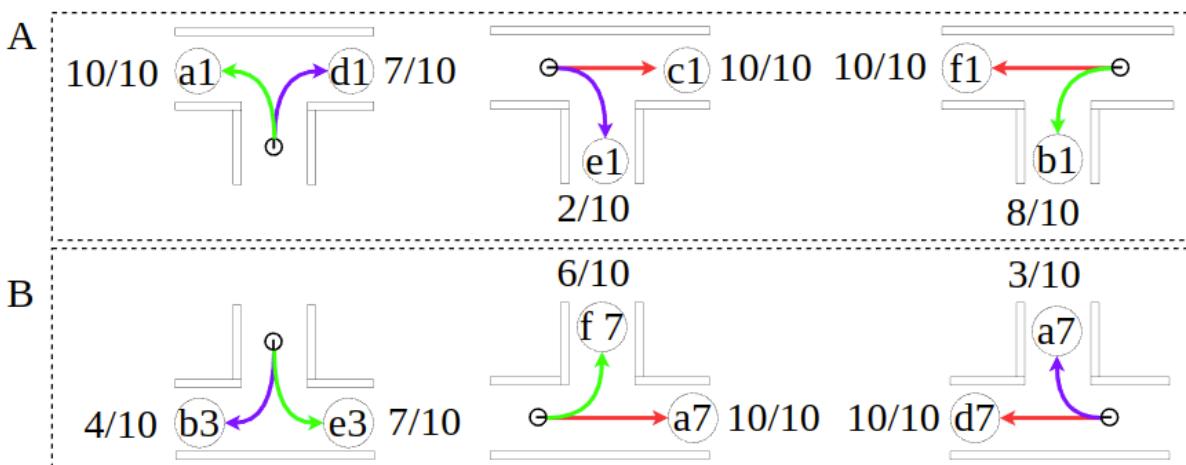
前章で述べたように、簡易的なシミュレータ上での実験により提案手法が有効だと確認されている。そのため、次の段階として、実環境における提案手法の有効性を検証することを試みた。しかし、学習時間の長さが問題となり、評価を十分な回数行うことが困難であった。そのため、本章では学習時間の短縮化に関して議論する。2つのアプローチを提案し、学習時間の削減の効果を実験により確認する。

## 5.2 学習 step 数の削減の影響

前章のシミュレータ上での実験と同様の条件で、単に 60000step から 10000step にステップ数を変更し、実験した結果を下記に示す。

実験結果を Fig. 5.1 に示す。この図は、それぞれの走行パターンにおいて正しく経路を選択し、走行できた回数を表している。Table 5.1 に実験ごとに全パターンの成功回数を合計した結果を示す。Table 5.1 に示すように、目標方向に従って 87/120 回、正しい経路を選択する様子が見られた。

60000step の実験と成功率を比較すると約 22% ほどの差がある。これまで述べた提案手法では、ステップ数を減らすことにより、成功率が大幅に低下することが判明した。そのため、後述する 2 つのアプローチを試みることで成功率の改善を図る。



○:Robot ○:Goal →:Go straight →:Turn left →:Turn right

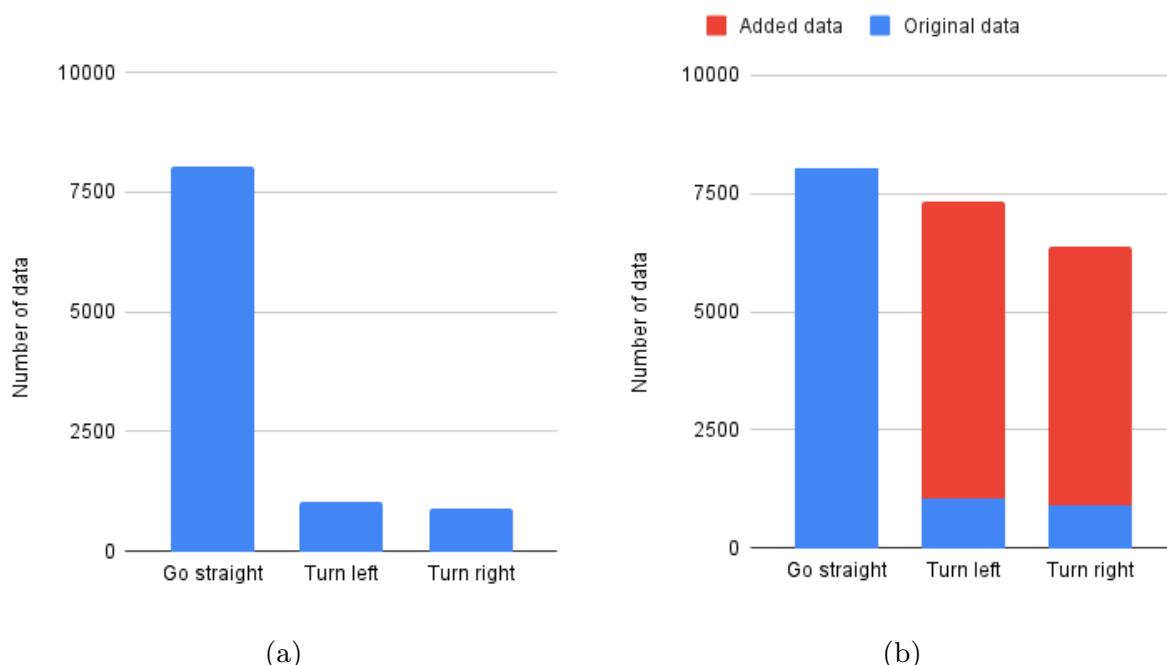
Fig. 5.1: Experimental results for each moving pattern at 10000step

Table 5.1: Experimental results at 10000step

Step	Total results
60000	113/120 (94.2%)
10000	87/120 (72.5%)

### 5.3 アプローチ 1：データセットに加えるデータの割合の変更

実験における 10000step ごとのコマンドのデータ数を Fig. 5.2 に示す。Fig. 5.2 (a) より、直進コマンド時のデータ数が圧倒的に多いことがわかる。このデータ数の偏りを解消するため、Fig. 5.2 (b) に示すように、左折と右折コマンドのデータ数を 7 倍にする。7 倍にするのは、左折と右折コマンドのデータ数を直進コマンドと同程度にするためである。なお、データ数を何倍にするのが望ましいのか本論文では議論しない。実験により、データセットに加えるデータ数の割合の変更が有効か検証する。なお、第3章で述べた提案手法は以後 CIL(Conditional Imitation Learning) と呼び、本アプローチを加えた手法を CIL with CoDP(Change of Data Percentage) と呼ぶ。



**Fig. 5.2:** Number of data per command per 10000steps in conventional experiments

#### 5.3.1 実験目的

データセットに加えるデータの割合変更が学習 step 数の削減に有効か検証を行う

### 5.3.2 実験装置

4.2で述べた簡易的なシミュレータ環境とロボットで実験を行った

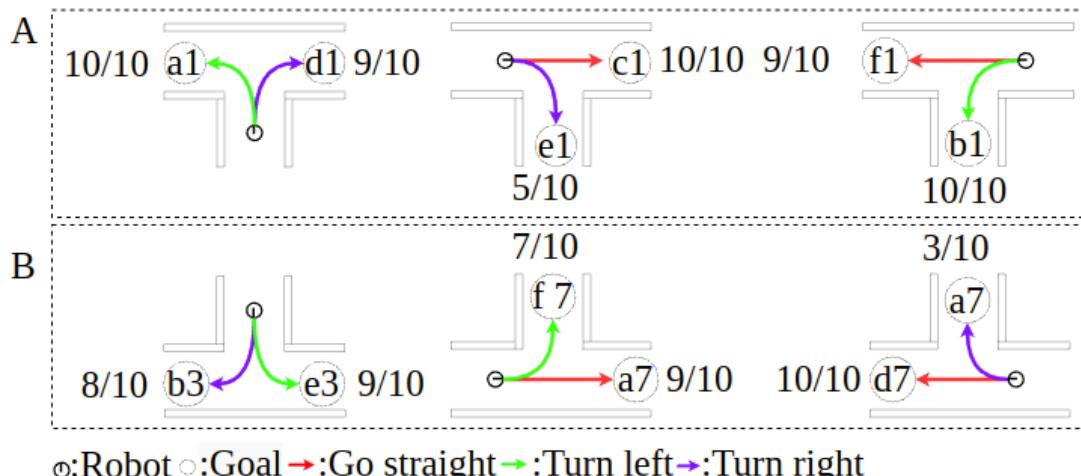
### 5.3.3 実験方法

4.3で示した経路を繰り返し走行させる。学習を10000step実行後、テストフェーズに移行する。テストフェーズで正しい順序で経路を選択し、走行を行えるか確認する。この一連の流れを10回繰り返し行う。

### 5.3.4 実験結果

実験結果をFig. 5.3に示す。この図は、それぞれの走行パターンにおいて正しく経路を選択し、走行できた回数を表している。Table 5.2に実験ごとに全パターンの成功回数を合計した結果を示す。Table 5.2に示すように、目標方向に従って99/120回、正しい経路を選択する様子が見られた。

前節の単にステップ数を減らすことに比べると成功率は高いが、60000stepの実験と成功率を比較すると約12%ほど差があり、アプローチ1を試みた結果では成功率が十分だとは言い難い。次に成功率を改善するため、後述するアプローチ2を試みた。



**Fig. 5.3:** Experimental results for each moving pattern by CIL with CoDP

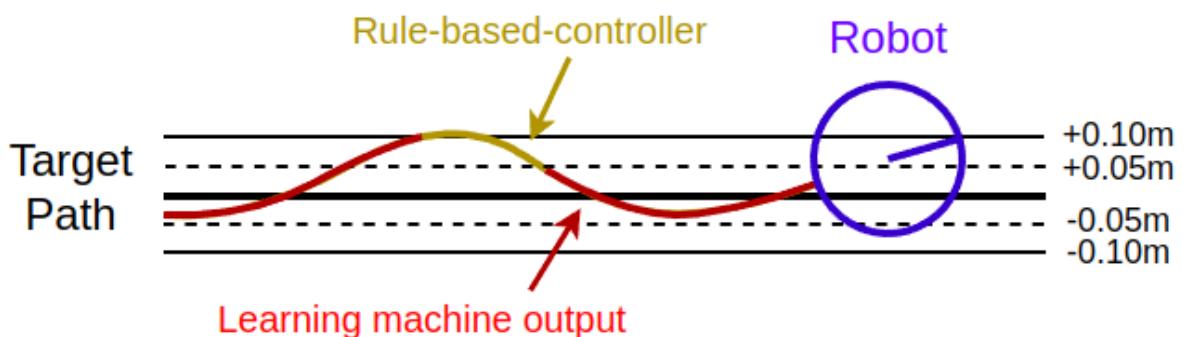
**Table 5.2:** Experimental results by CIL with CoDP

Category	Step	Total results
CIL	60000	113/120 ( 94.2% )
CIL	10000	87/120 ( 72.5% )
CIL with CoDP	10000	99/120 ( 82.5% )

## 5.4 アプローチ 2：学習フェーズにおける積極的な蛇行

清岡ら [13] により、目標経路上に加えて離れた状態を学習することが、テストフェーズでの走行に大きな影響を与えるため、重要だとされている。そのため、より積極的に蛇行を行い、目標経路から離れた状態を増加させることを検討する。

実験で用いているシステムの学習フェーズでは、Fig. 5.4 に示すようにロボットが目標経路上付近を走行している場合、訓練中の学習器へ画像を入力し、出力される角速度を用いて走行している。この場合に、目標経路から一定の距離離れると地図を用いたルールベース制御器の走行に切り替わり、強制的にロボットを目標経路上に戻すように制御を行う。

**Fig. 5.4:** Moving on the target path

訓練中の学習器へ画像を入力し、Fig. 5.5 のように出力される角速度を 1.5 倍にする。その結果、Fig. 5.6 に示すように蛇行する頻度が高くなり、目標経路から離れた状態をより多く学習できる可能性がある。なお、得られた角速度を何倍にするのが望ましいのか本論文では議論しない。実験により、学習フェーズにおける積極的な蛇行が有効か検証する。なお、CIL

with CoDP に本アプローチを加えた手法を CIL with CoDP+AM(Aggressive Meandering) と呼ぶ。

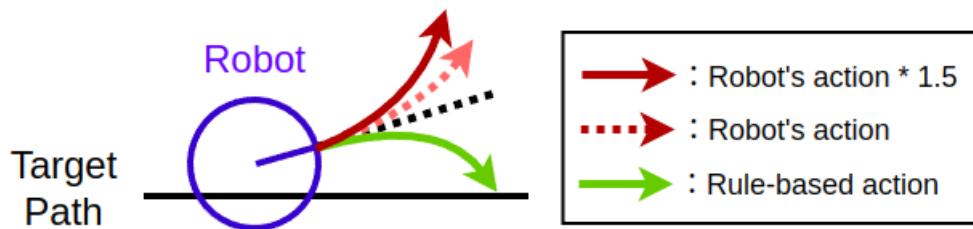


Fig. 5.5: Robot behavior

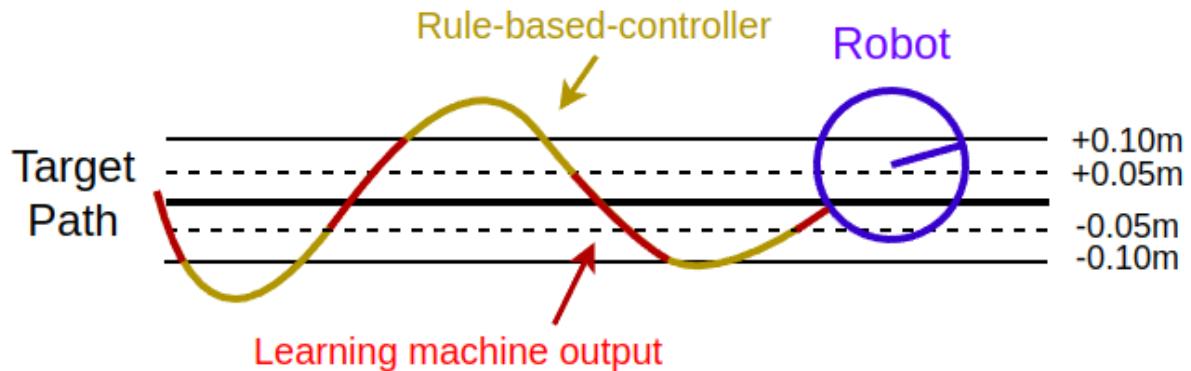


Fig. 5.6: Moving on the target path attempting CIL with CoDP+AM

#### 5.4.1 実験目的

学習フェーズにおける積極的な蛇行が学習 step 数の削減に有効か検証を行う

#### 5.4.2 実験装置

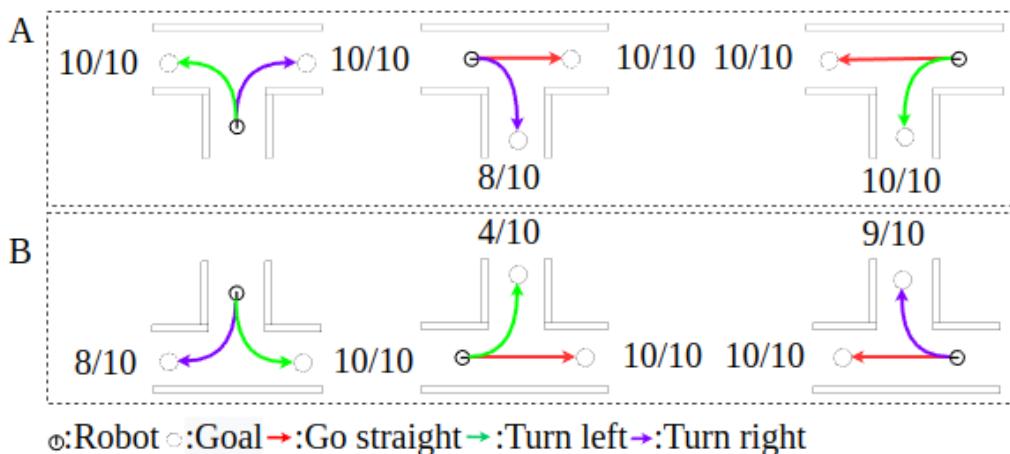
4.2 で述べた簡易的なシミュレータ環境とロボットで実験を行った

#### 5.4.3 実験方法

4.3 で示した経路を繰り返し走行させる。学習を 10000step 実行後、テストフェーズに移行する。テストフェーズで正しい順序で経路を選択し、走行を行えるか確認する。この一連の流れを 10 回繰り返し行う。

#### 5.4.4 実験結果

実験結果を Fig. 5.7 に示す。この図は、それぞれの走行パターンにおいて正しく経路を選択し、走行できた回数を表している。Table 5.3 に実験ごとに全パターンの成功回数を合計した結果を示す。Table 5.3 に示すように、目標方向に従って 109/120 回、正しい経路を選択する様子が見られた。



**Fig. 5.7:** Experimental results for each moving pattern at 10000step by CIL with CoDP+AM

**Table 5.3:** Experimental results at 10000step by CIL with CoDP+AM

Category	Step	Total results
CIL	60000	113/120 ( 94.2% )
CIL	10000	87/120 ( 72.5% )
CIL with CoDP	10000	99/120 ( 82.5% )
<b>CIL with CoDP+AM</b>	<b>10000</b>	<b>109/120 ( 90.8% )</b>

Table 5.3 からアプローチ 1 の実験より、成功率が改善していることがわかる。また、60000step の実験と成功率を比較した場合、約 3% ほどの差がある。

以下に Fig. 5.7 における成功率が低い場所と要因を示す。

- f7

4.3 で示した経路の最後であり、学習フェーズが終了する直前の三叉路である。そのた

め, データセットに f7 のデータが加えられてから十分な学習ができていない

- a7, b3, e1

これらは, 全て右折する場所である. Fig. 5.2 より, 他のコマンドのデータと比較して右折コマンドのデータが少ないことがわかる. このことから, 右折コマンドのデータを十分に学習できていない

この 2 つの点から, 学習量を増加すると成功率が改善する可能性がある. そのため, 10000step から 20000step に学習量を増加した実験結果を後述する.

学習フェーズにおける目標経路からの距離によるデータの割合を Fig. 5.8 に示す。

この図が示すように、学習器から得られた角速度を 1.5 倍にした場合、データの割合が目標経路付近では減少し、目標経路から離れた場所では増加している。よって、アプローチ 2 を試みた結果、学習フェーズでは目標経路から離れる行動が増えた。すなわち、積極的に蛇行している可能性が高い。

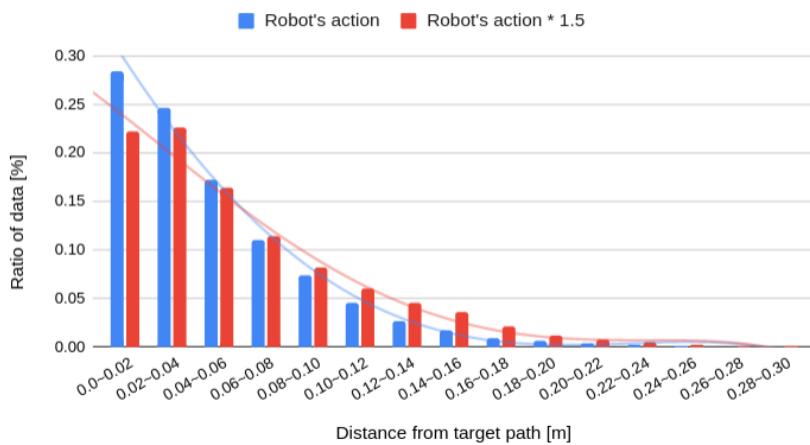


Fig. 5.8: Ratio of data by distance from target path in learning phase

テストフェーズにおける目標経路からの距離によるデータの割合を Fig. 5.9 に示す。この図から、学習器から得られた角速度を 1.5 倍にした場合、データの割合が目標経路付近では増加し、目標経路から離れた場所では減少している。よって、アプローチ 2 を試みた結果、テストフェーズでは目標経路付近の行動が増えた。すなわち、アプローチ 2 を試みる前に比べ、より正確に経路追従行動を模倣している可能性が高い。

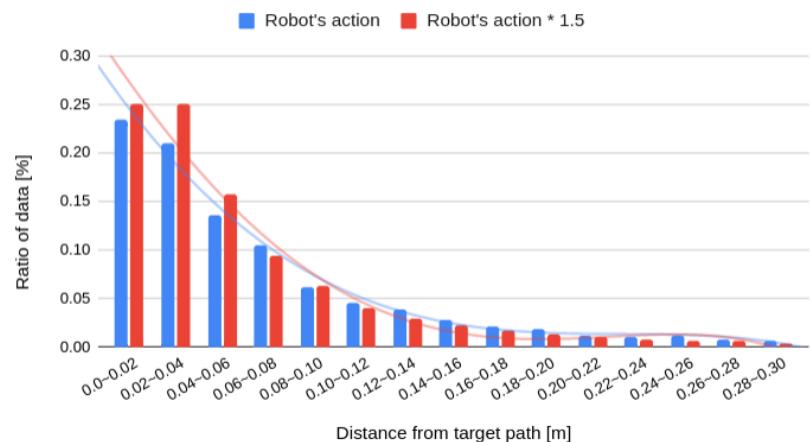


Fig. 5.9: Ratio of data by distance from target path in test phase

CIL with CoDP+AM で、10000step から 20000step に学習ステップ数を増やし、実験した結果を下記に示す。

Fig. 5.10 は、それぞれの走行パターンにおいて正しく経路を選択し、走行できた回数を表している。Table 5.4 に実験ごとに全パターンの成功回数を合計した結果を示す。Table 5.4 に示すように、目標方向に従って 114/120 回、正しい経路を選択する様子が見られた。また、成功率が 60000step の実験と同程度になった。このことから、提案する CIL with CoDP+AM によって学習のステップ数が約 67% 削減できたといえる。

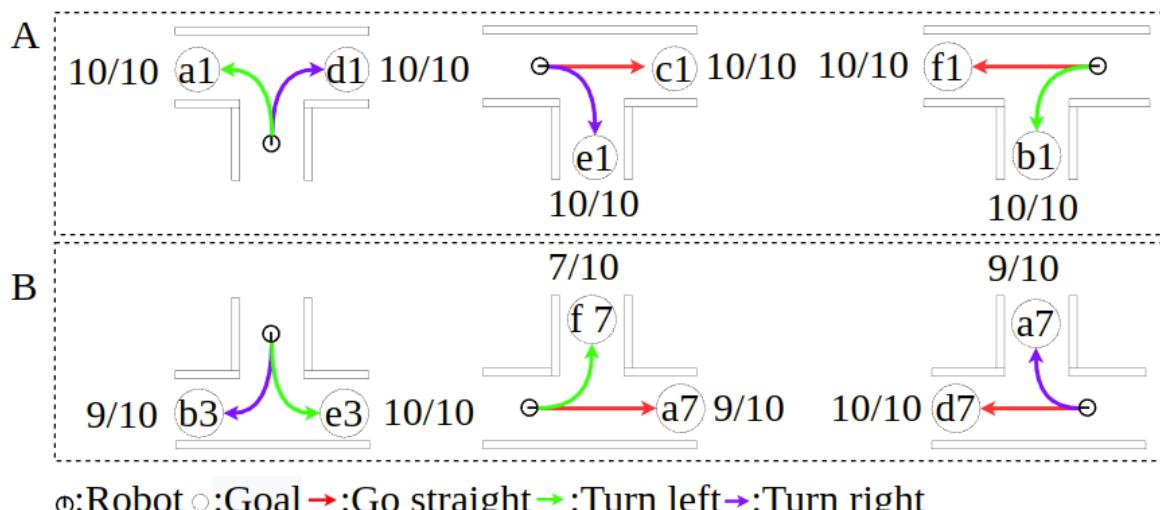


Fig. 5.10: Experimental results for each moving pattern at 20000step by CIL with CoDP+AM

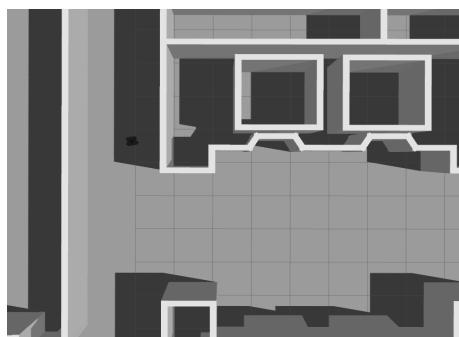
Table 5.4: Experimental results at 20000step by CIL with CoDP+AM

Category	Step	Total results
CIL	60000	113/120 ( 94.2% )
CIL	10000	87/120 ( 72.5% )
CIL with CoDP	10000	99/120 ( 82.5% )
CIL with CoDP+AM	10000	109/120 ( 90.8% )
<b>CIL with CoDP+AM</b>	<b>20000</b>	<b>114/120 ( 95% )</b>

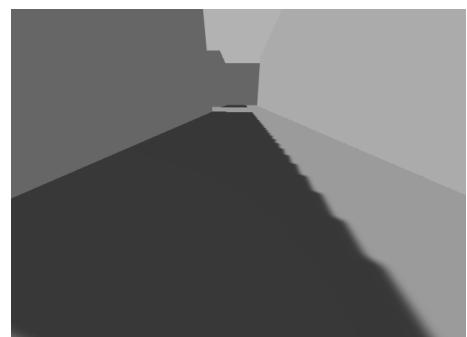
## 5.5 実環境に似たシミュレータ環境による実験

これまでの実験では、簡易的なシミュレータ環境を用いてきた。しかし、実験に簡易的なシミュレータ環境を用いるには問題があり、以下のような問題がある。

- Fig. 5.11 (a) に示すように、環境の大半が灰色や白のみで構成されているため、Fig. 5.11 (b) のように視覚による特徴が乏しい。



(a) A bird's eye view of the robot



(b) Robot Perspective

**Fig. 5.11:** Simple simulator environment

この問題は、学習フェーズにおいて取得するカメラ画像で学習器を訓練する際に、大きな影響を及ぼす可能性がある。そのため、ロボットの視覚であるカメラ画像で、より多くの視覚的特徴をとらえるために、Fig. 5.12 に示すように実環境に似たシミュレータ環境を作成した。



(a) A bird's eye view of the robot



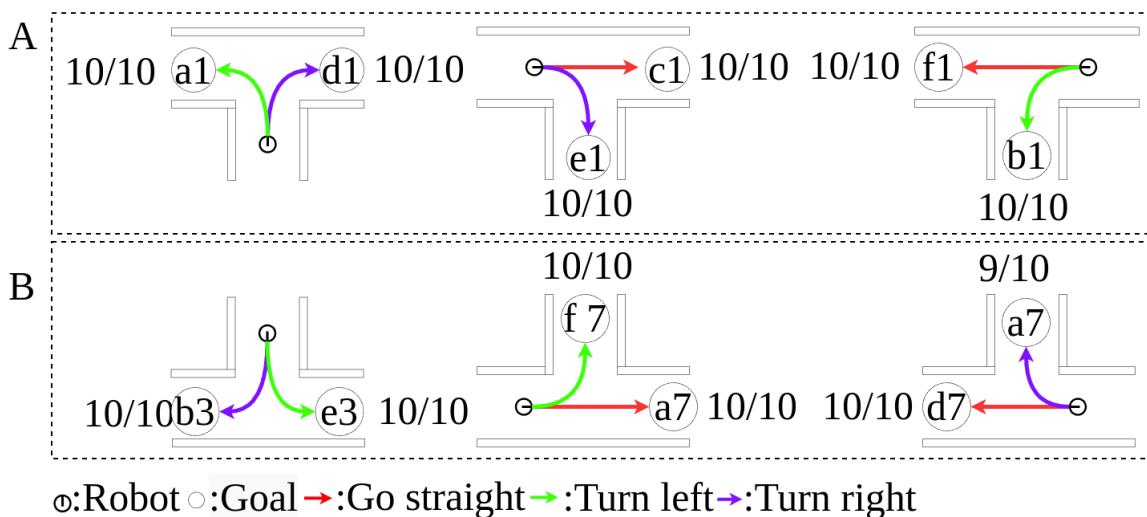
(b) Robot Perspective

**Fig. 5.12:** Simulator environment similar to real environment

上記の実環境に似せたシミュレータ環境で、5.4章の CIL with CoDP+AM と同じ条件で、20000step 実験した結果を下記に示す。

Fig. 5.13 は、それぞれの走行パターンにおいて正しく経路を選択し、走行できた回数を表している。Table 5.5 に実験ごとに全パターンの成功回数を合計した結果を示す。Table 5.5 に示すように、目標方向に従って 119/120 回、正しい経路を選択する様子が見られた。

簡易的なシミュレータ上での実験と比較すると、成功率が改善しており、限りなく 100% に近づいている。これは、大半が灰色と白で構成されている簡易的なシミュレータとは異なり、実環境に似たシミュレータでは、よりロボットが捉えることのできる特徴（床の模様、壁と床の境界線など）が増えたためだと考えられる。



**Fig. 5.13:** Experimental results for each moving pattern at 20000step by Approach 1+2(simulator environment to real environment)

**Table 5.5:** Experimental results by simulator similar to real environment

Environment (CIL with CoDP+AM)	Step	Total results
Simple simulator environment	20000	114/120 ( 95% )
<b>Simulator similar to real environment</b>	<b>20000</b>	<b>119/120 ( 99.2% )</b>

## 5.6 実環境の実験

これまで、実験の大半をシミュレータ上で行ってきたが、実験を実環境に移す。本節では、実環境における提案手法の有効性を検証する。

### 5.6.1 実験目的

実環境において、提案手法が有効か検証を行う

### 5.6.2 実験装置

- ロボット

ロボットは従来手法の検証に用いた実験 [2] と同様、Fig. 5.14 に示すように、3つのカメラを搭載したロボットを用いる。

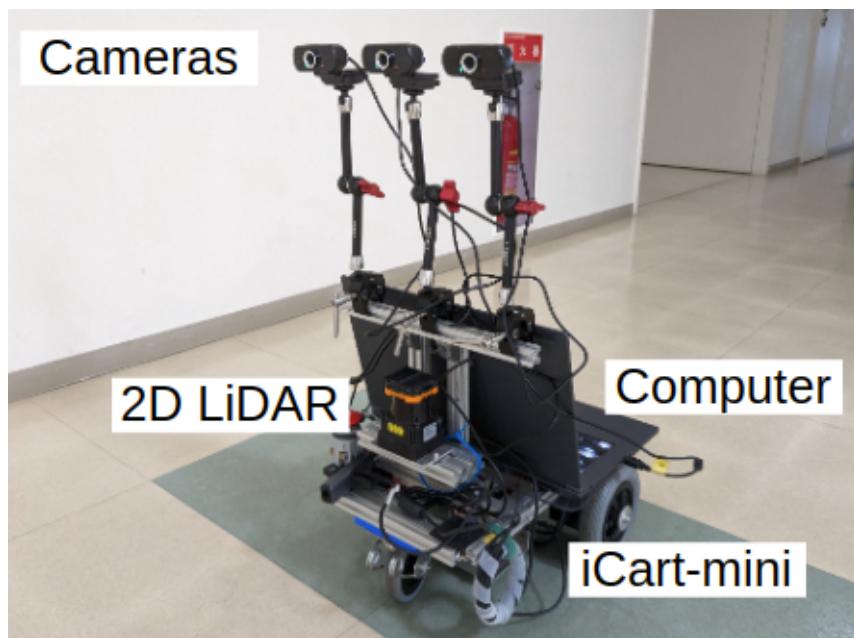


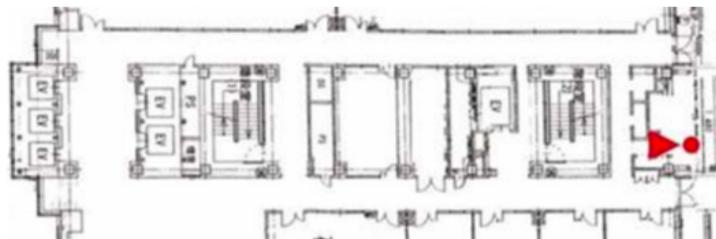
Fig. 5.14: Experimental setup

- 環境

Fig. 5.15 に示すような千葉工業大学津田沼キャンパス 2号館 3階で実験を行う。



(a) One place in the real environment



(b) structure

Fig. 5.15: Real environment

### 5.6.3 実験方法

5.5 章の実験により、実環境の実験においても 20000step ほど学習させることで、高い成功率が得られる可能性が高いことが予想される。しかし、実環境で行う実験時間を削減するため、実環境に似たシミュレータ上でまずは訓練を行い、その学習器を実環境でファインチューニングする。なお、本論文ではファインチューニングによる実験結果の影響を議論しない。実環境における実験の流れを以下に示す。

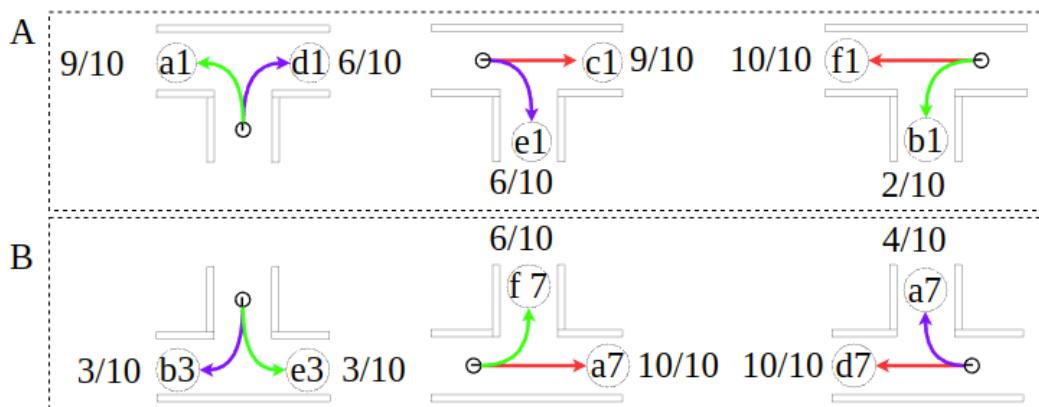
1. 事前に 5.5 章の実験に倣って、シミュレータ上で 10000step 学習させる。
2. この学習器の重みを使って、実環境において 4.1.2 で示した経路を繰り返し走行せながら追加で学習を行う。
3. 学習を 10000step 実行後、テストフェーズに移行する。テストフェーズで正しい順序で

経路を選択し、走行を行えるか確認する。

この一連の流れを 10 回繰り返し行う。

#### 5.6.4 実験結果

実験結果を Fig. 5.16 に示す。この図は、それぞれの走行パターンにおいて正しく経路を選択し、走行できた回数を表している。Table 5.6 に実験ごとに全パターンの成功回数を合計した結果を示す。Table 5.6 に示すように、目標方向に従って 78/120 回、正しい経路を選択する様子が見られた。



**Fig. 5.16:** Experimental results for each moving pattern by real environment

**Table 5.6:** Experimental results by real environment

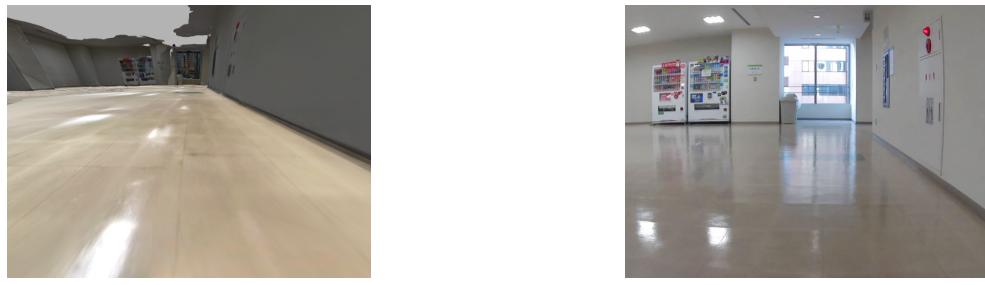
Environment(CIL with CoDP+AM)	Step	Total results
Simulator similar to real environment	20000	119/120 ( 99.2% )
<b>Real environment</b>	<b>20000</b>	<b>78/120 ( 65% )</b>

実環境における実験の成功率が明らかに低い結果となった。このような結果になった原因について、考えられるのは以下の 3 点である。

1. データセットに加えたカメラ画像に人などが多く写ってしまった
2. データセットに加えたカメラ画像の光量が変化してしまった
3. 画像を取得するカメラの視野角と高さがシミュレータと実環境で違う

上記の 1,2 に関して、シミュレータ上での学習では環境中の光や障害物の有無は変化しない。一方で、実環境での実験では通行人の存在や僅かな光の変化がある。

Fig. 5.17 はシミュレータ上と実環境で対応する場所で取得した画像を示している。この図より、カメラの視野角と高さが異なっていることがわかる。このことから、上記の 3 に関して同一の場所においても視野角の違いなどにより、取得できる画像に差があることが影響しているおそれがある。



(a) Simulator similar to real environment

(b) Real environment

Fig. 5.17: Acquired camera images at corresponding places

Fig. 5.18 は実環境における実験で、成功率が最も高いモデルによる実験結果である。9/12 回正しい経路を選択する様子が見られた。この図の b3 と e3 に向かう三叉路などでは、経路選択前は同じ場所を走行しているため、ほぼ同様の画像が入力されているが、目標方向に従って正しい経路を選択できている。このことから、実環境においても提案手法により、任意の経路選択が行うことができている可能性が高い。しかし、成功率が低い原因の解明には未だ至っていない。この問題の検討は、今後の課題となっている。

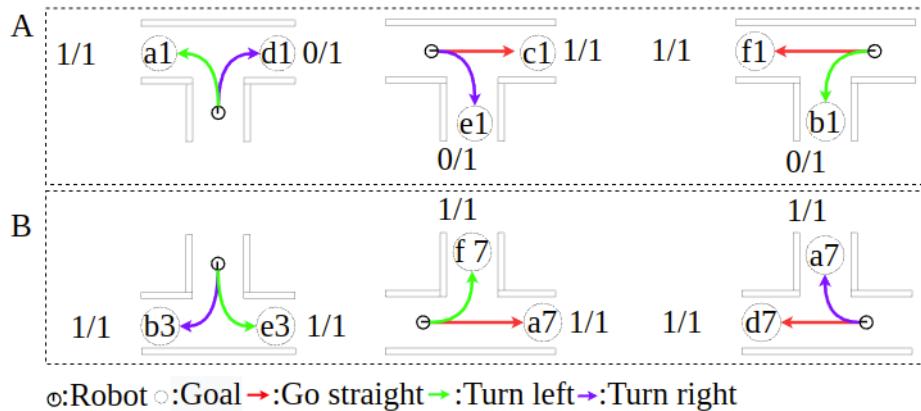


Fig. 5.18: Experimental results for each moving pattern by models with the highest success rate

## 第6章

### 結論

本研究では、経路追従行動をカメラ画像を用いた end-to-end 学習で模倣する岡田らの従来手法をベースに、データセットと学習器の入力へ目標方向を加えることで、経路選択をする機能の追加を提案した。また、シミュレータ上での実験を実環境に移す際に、問題となった学習時間の長さについて、2つのアプローチを試みることで学習時間を大幅に削減した。加えて、実環境での実験を行い、有効性の検証を行った。実験結果より、学習器へ目標方向を与えることで、指定した経路へ走行する挙動が確認できた。

## 参考文献

- [1] Mariusz bojarski et al. "end to end learning for self-driving cars ". arxiv: 1604.07316, 2016.
- [2] 岡田眞也, 清岡優祐, 上田隆一, 林原靖男“ 視覚と行動の : end-to-end 学習により経路追従行動 をオンラインで模倣する手法の提案 ” , 計測自動制御学会 si 部門講演会 sice-si2020 予稿集,pp.1147–1152(2020).
- [3] Felipe codevilla et al. "end-to-end driving via conditional imitation learning ". arxiv: 1710.02410, 2018.
- [4] Jeffrey hawke et al. " urban driving with conditional imitation learning " . arxiv: 1912.00177, 2019.
- [5] I. sutskever a. krizhevsky and g. e. hinton. imangenet classification with deep convolutional neural networks. pp. 2p1-j07. 一般社団法人 日本機械学会, 2017.
- [6] 春山健太, 藤原恆, 清岡優祐, 岡田眞也, 上田隆一, 林原靖男, “ 視覚と行動の end-to-end 学習により経路追従行動をオンラインで模倣する手法の提案 経路選択機能の追加 ” , 日本機械学会ロボティクス・メカトロニクス講演会 ’ 22 予稿集,(2022).
- [7] 岡田眞也, 清岡優祐, 春山健太, 上田隆一, 林原靖男: “ 視覚と行動の end-to-end 学習により経路追従行動 をオンラインで模倣する手法の提案-経路追従行動の修正のためにデータセットを動的に追加する手法の検討 ” , 計測自動制御学会 si 部門講演会 sice-si2021 予稿集,pp.1066-1070(2021).
- [8] Karen simonyan et al. "very deep convolutional networks for large-scale image recognition " . arxiv:1409.1556v6, 2015.
- [9] ros-planning, navigation レポジトリ. <https://github.com/ros-planning/navigation>. (Accessed on 12/6/2022).

- [10] waypoint\_nav レポジトリ. [https://github.com/open-rdc/waypoint\\_nav.git](https://github.com/open-rdc/waypoint_nav.git). (Accessed on 12/6/2022).
- [11] Turtlebot3 robotis emanual.robotis. <https://emanual.robotis.com/docs/>. (Accessed on 12/31/2022).
- [12] gazebo. <http://gazebosim.org/>. (Accessed on 12/31/2022).
- [13] 春山健太, 藤原柾, 清岡優祐, 岡田眞也, 上田隆一, 林原靖男, “視覚と行動の end-to-end 学習により経路追従行動をオンラインで模倣する手法の提案 経路選択機能の追加”, 日本機械学会ロボティクス・メカトロニクス講演会 ’22 予稿集,(2022).

# 付録

## 動画

実験の結果から得られたモデルを用いて、中間層の可視化を行った状態で学習器の出力において走行させた際の動画を記録した。CNN が通路の特徴（輪郭、または壁と床の境界線）を捉えるように学習を行ったことが見て取れる。以下にアップロードした動画の URL を掲載する。

<https://youtu.be/JusGVH6ejFg>

## 謝辞

本研究を進めるにあたり，1年に渡り，熱心にご指導を頂いた林原靖男教授に深く感謝いたします．ロボット設計制御研究室の皆様には，ご意見，ご協力頂き感謝申し上げます．特に，春山健太氏には，研究の着想や実験など多くの助言をして頂きましたことを心より感謝いたします．