シミュレーションと強化学習に基づく移動障害物回避機能を持つ自律移動ロボット

背景(1/2)

- 自律移動ロボットの需要増加
  - 配膳や配達などの労働力不足の解決
  - 人の移動手段の代替
- 屋内や歩道を走行
- 人が存在する環境で安全な走行が必要

自律移動ロボットにおいて
人の回避は必須の課題

BellaBot [1]

WHILL Model C2 [2]

[1]テクノホライゾン株式会社. BellaBot.
https://www.elmo.co.jp/product/robot/bellabot/
[2] WHILL株式会社. WHILL Model C2. https://whill.inc/jp/

背景(2/2)

- ルールベース制御が主流
- 人回避における現状の課題
  - 効率や汎用性に問題あり
    - ✓ ルールベース制御は人の行動の途中変化に対応困難
    - ✓ 静止して人の回避行動を待つのは非効率
- 実環境における実験実施の問題
  - 衝突の危険性
  - 多様な状況の再現の困難

仮想環境で学習ベースの
移動障害物回避を検討

目的とアプローチ

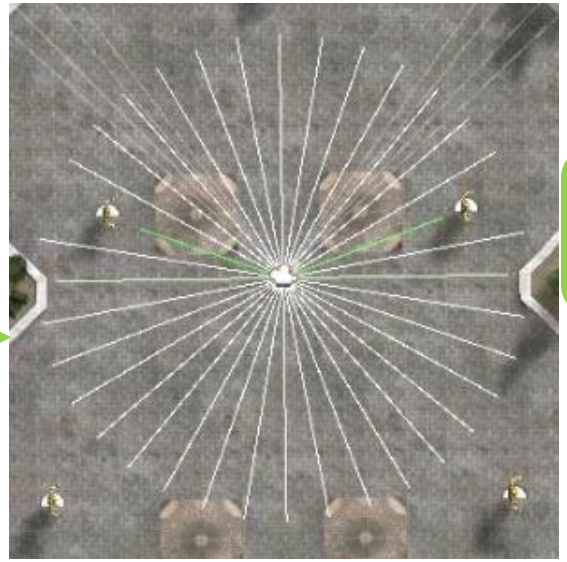
- 目的
  - 移動障害物回避が可能な自律走行システムの構築
- アプローチ
  - 移動障害物回避のシミュレーション環境の作成
  - 強化学習を用いた移動障害物回避行動の獲得
  - ルールベース制御モデルと学習モデルの精度比較

使用するセンサ

- 距離センサ

LiDAR[3]

シミュレーション



Raycast Sensor

- ビジョンセンサ

グレースケールカメラ[4]

シミュレーション

グレースケールカメラ

[3] Suteng Innovation Technology Co. Ltd, 3D LiDAR
https://www.zmp.co.jp/products/sensor/3d-lidar/rslidar
[4]株式会社ロジクール, RGBカメラ
https://www.logicool.co.jp/ja/jp/products/webcams.html

シミュレーション環境の作成(1/2)

- 環境
  - ショッピングモールの３Dモデル
- 移動障害物
  - 人の３Dモデル

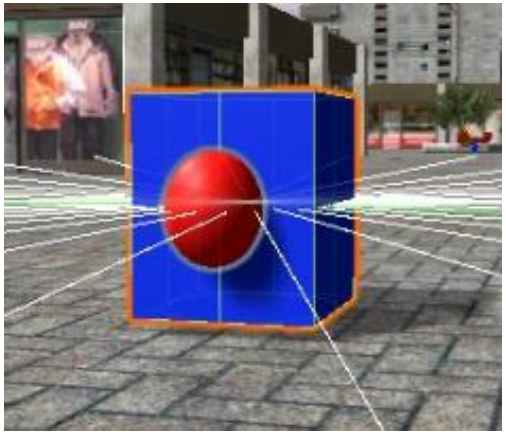
Shopping Mall HQ [5]

StarterAssets thirdPerson [6]

[5] Unity Technologies, Unity Asset Store Shopping Mall HQ, https://assetstore.unity.com/packages/3d/environments/urban/shopping-mall-hq-2293
[6] Unity Technologies, Unity Asset Store StarterAssets thirdPerson, https://assetstore.unity.com/packages/essentials/starter-assets-third-person-character-controller-196526

シミュレーション環境の作成(2/2)

- 自律移動ロボットの３Dモデル
  - 形状は立方体
  - 進行方向を示す球体
- 距離センサ
  - 360度計測
  - 近づく障害物を認識
- ビジョンセンサ
  - 進行方向の
    2次元グレースケール画像
    - ✓ 色情報を除外
  - 障害物の形や大きさを認識

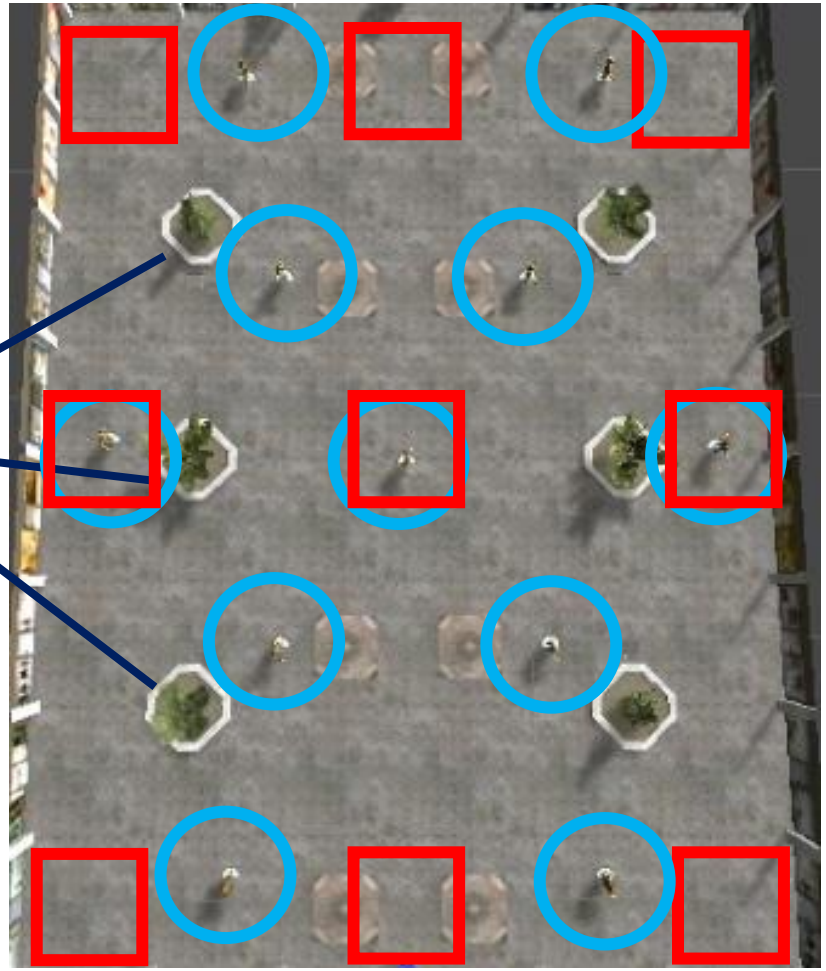


自律移動ロボットの3Dモデル

グレースケール画像

移動障害物の設定(1/3)

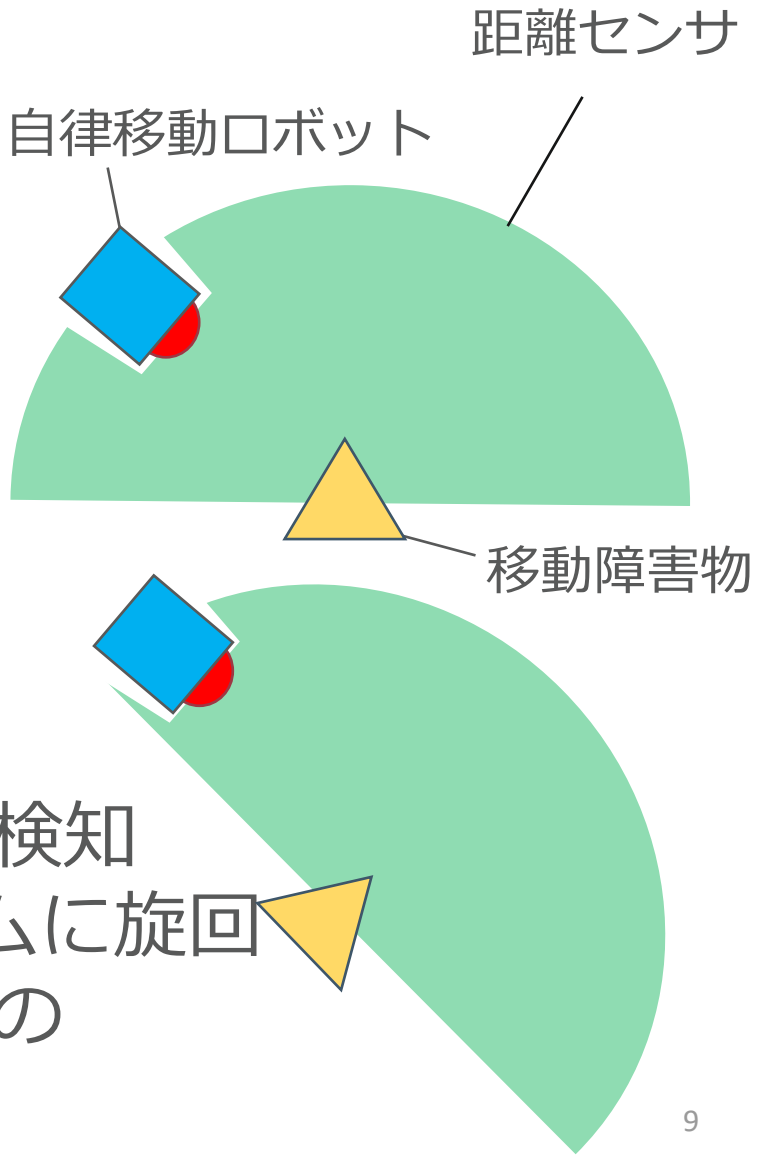
- 移動速度
  - 自律移動ロボットの速度以下のランダム値
  - 一定の速度
- 初期位置
  - 図の青色の円
- 挙動
  - 図の赤色のエリア間を自由に行き来



移動障害物の初期位置

移動障害物の設定(2/3)

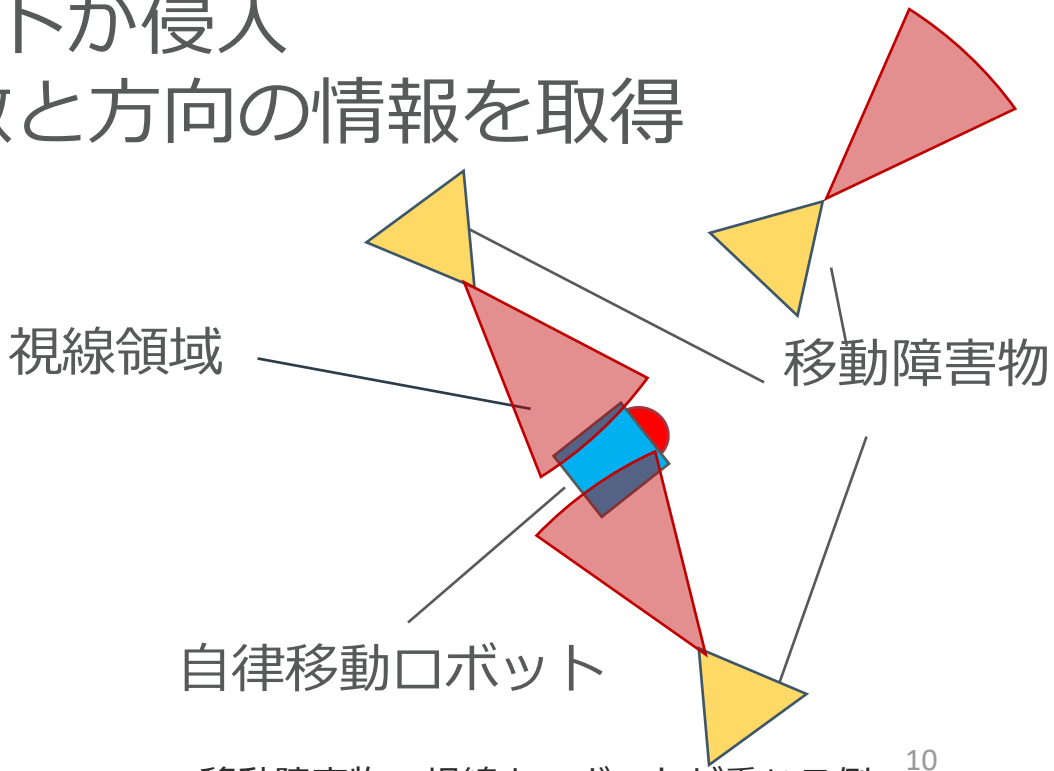
- 移動障害物の挙動
  - 距離センサで自律移動ロボットの接近を認識
  - 目的地への移動を中断
  - 静止・目的地への移動・回避行動を等確率で決定



- 移動障害物の回避行動
  - 自律移動ロボットの方向を検知
  - 逆方向に0~90度でランダムに旋回
  - しばらく前進し、目的地への移動を再開

移動障害物の設定(3/3)

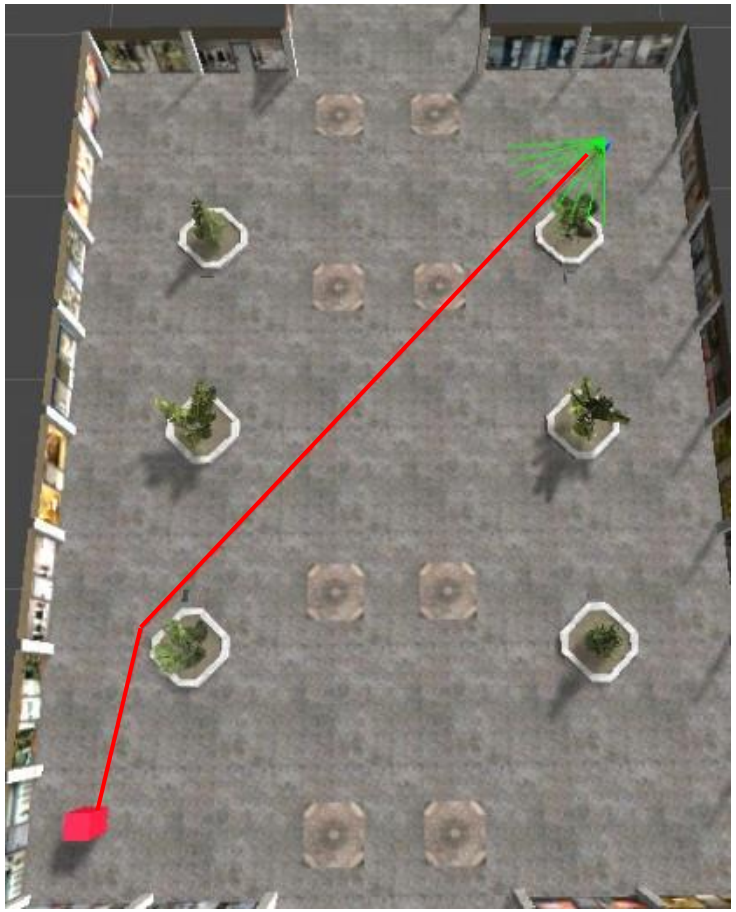
- 移動障害物の視線領域
  - ロボットを見る移動障害物の情報
  - ロボットが観測データとして使用
  - 視線領域にロボットが侵入
    - ✓ 移動障害物の数と方向の情報を取得



移動障害物の視線とロボットが重なる例

目的地への走行

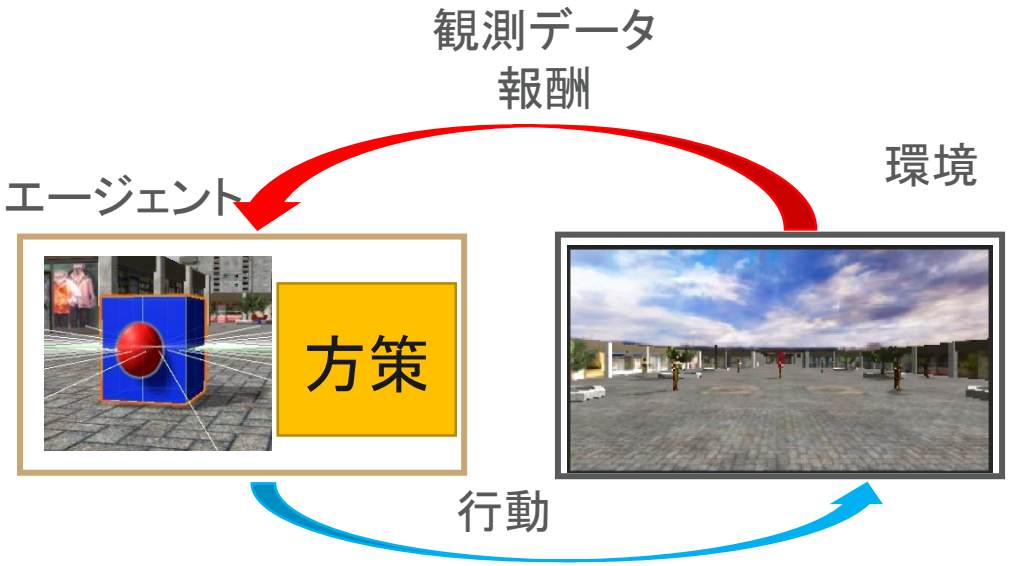
- NavmeshAgent[7]
  - 出発地点から目的地点までの最短経路生成（赤線）
  - A*探索アルゴリズム[8]による経路探索
  - 移動障害物，自律移動ロボットに設定



自律移動ロボットの経路生成の様子

[7] Unity Technologies, Unity Documentation NavMesh Agent, https://docs.unity3d.com/2021.3/Documentation/Manual/class-NavMeshAgent.html
[8] P. E. Hart N. J. Nilsson and B. Raphael, “A Formal Basis for the Heuristic Determination of Minimum Cost Paths,” IEEE Transactions on Systems Science and Cybernetics, pp. 100-107, 1968

強化学習



- 観測データを使用して行動を決定し
  行動に応じた報酬を取得
- 方策
  - 観測データを入力し, 行動を出力する関数
- 経験
  - 観測データと行動と報酬の組み合わせ

経験に応じて報酬和を最大化するように方策を更新

報酬の設定

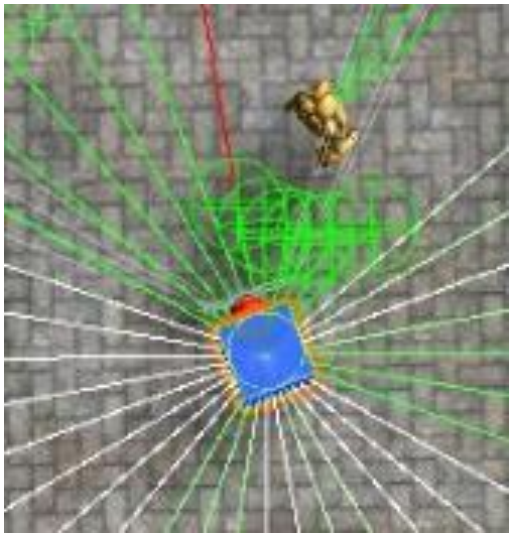
- 正の報酬
  - 目的地に到達
  - 移動障害物後方の半円領域に接触
    - ✓ 回避行動の学習を促進

- 負の報酬
  - 移動障害物に接触
  - 移動障害物前方の長方形領域に接触
    - ✓ 人の前方通過を抑制
  - 毎ステップ
    - ✓ 最短経路の学習を促進

学習モデルの詳細(1/2)

- 入力（観測データ）
  - 自律移動ロボットの速度, 向き
  - グレースケール画像
  - 距離センサ
  - 目的地の経由点への向き
  - 移動障害物の視線情報
- ２種類のモデル
  - 「移動障害物の視線情報なし」
  - 「移動障害物の視線情報あり」

グレースケール画像



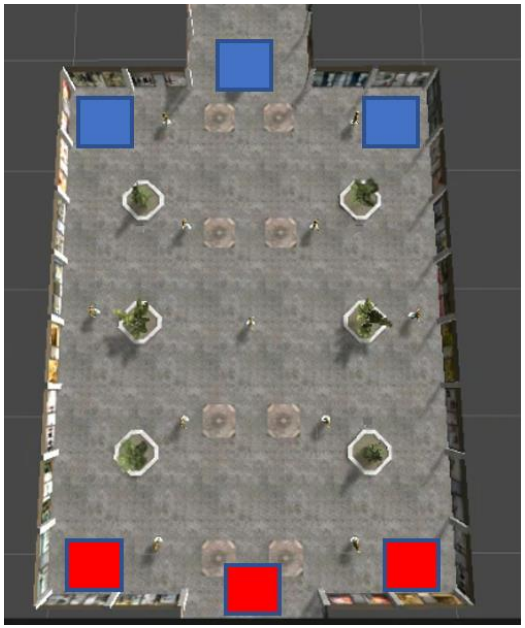
距離センサ

学習モデルの詳細(2/2)

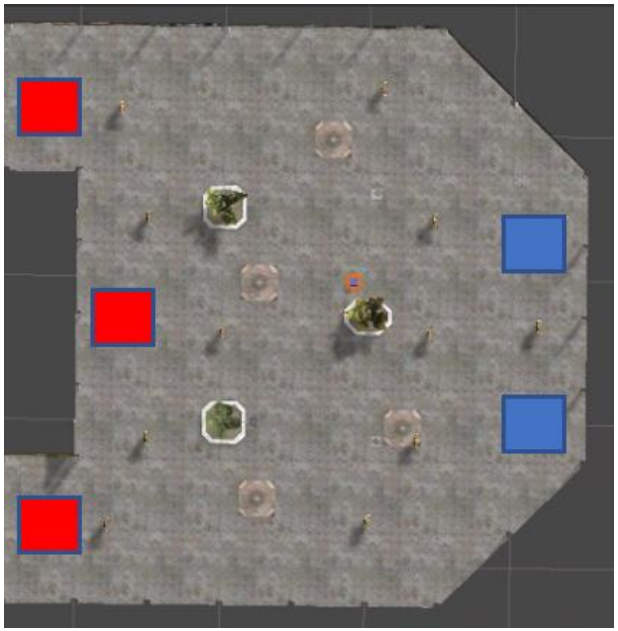
- 出力（行動）
  - 自律移動ロボットの速度
    - ✓ 前進・前進の半分・停止
  - 自律移動ロボットの向き
    - ✓ 変更なし・右旋回・左旋回
  - １ステップ(0.02秒)毎に決定

モデルの学習(1/2)

- 自律移動ロボットと目的地の初期位置
  - ➢ 異なる色の組み合わせの位置から
    ランダムに決定
    - ✓ 組み合わせ間の距離の差をなくす
- モデルの学習
  - ➢ 学習環境
- 精度の検証
  - ➢ 評価環境



学習環境



評価環境

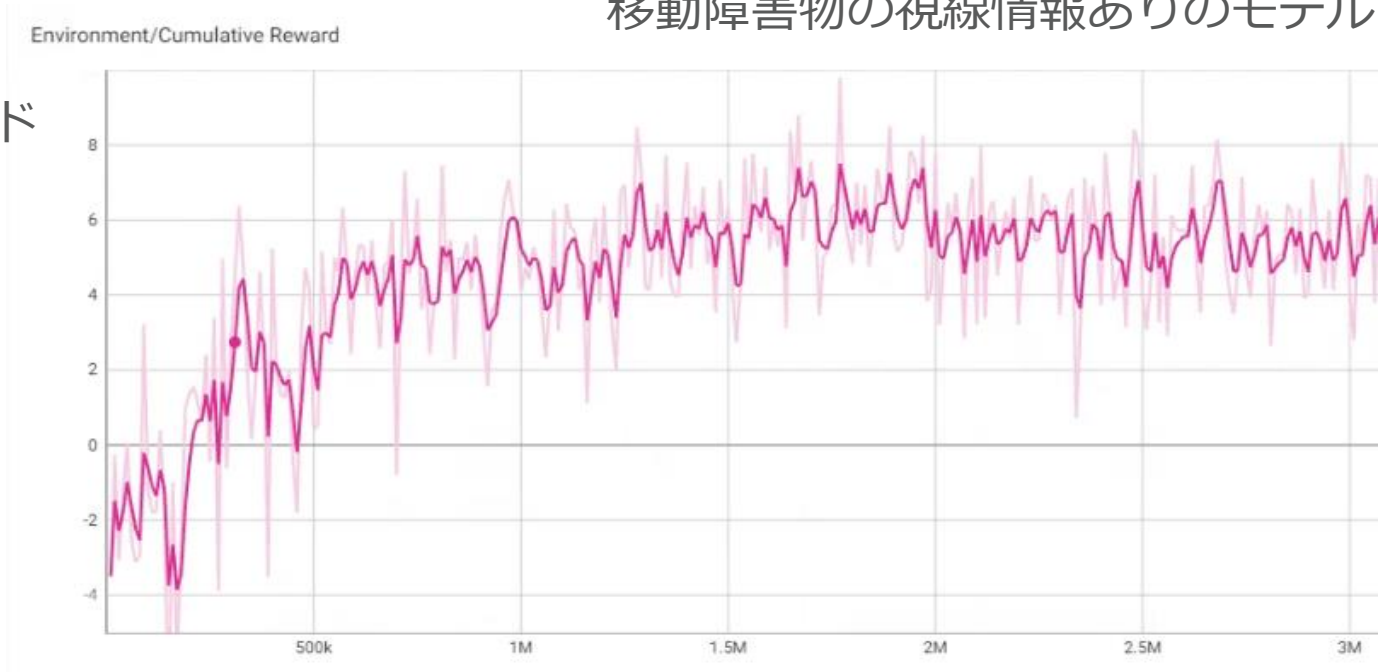
モデルの学習(2/2)

- 学習は300万ステップ
- 1エピソードは最大6000ステップ
  - 遠回り，長時間の静止の抑制
- エピソード終了条件
  - 目的地に到達
  - 移動障害物に衝突

学習結果

- 1万ステップ当たりの平均累積エピソード報酬
  - 100万ステップまで増加
  - 100万~300万ステップで小さな増減

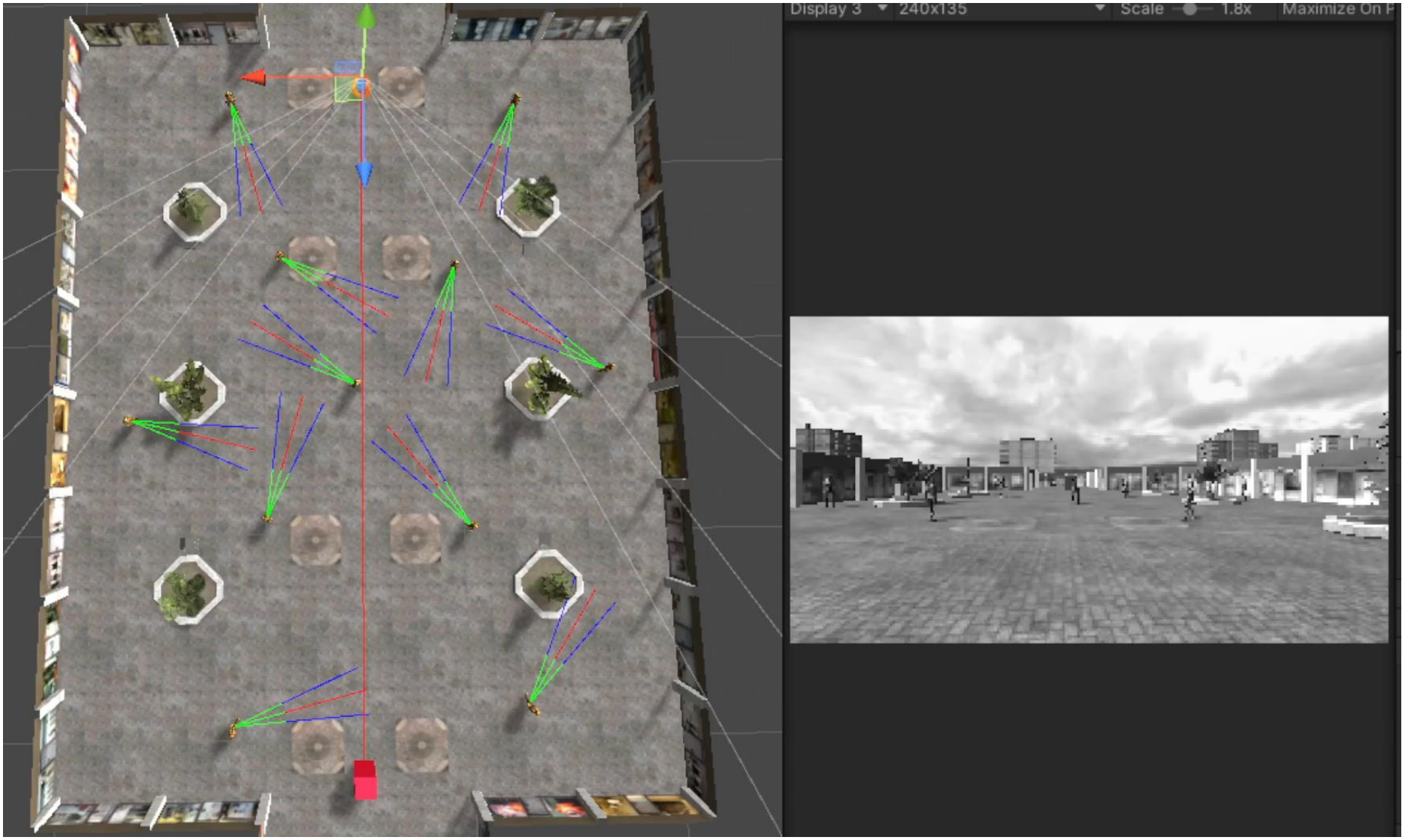
移動障害物の視線情報ありのモデル



平均累積エピソード報酬

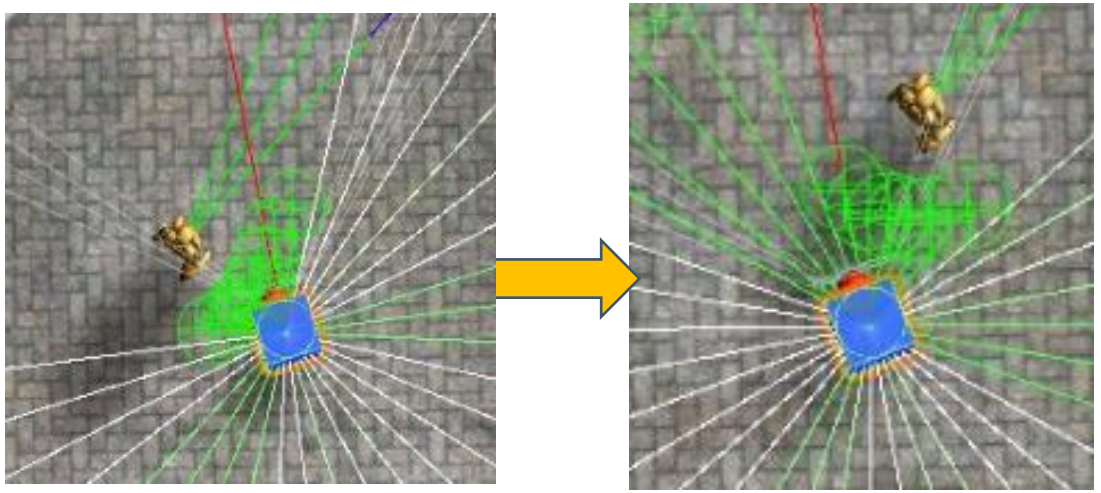
ステップ数

学習モデルの実行例



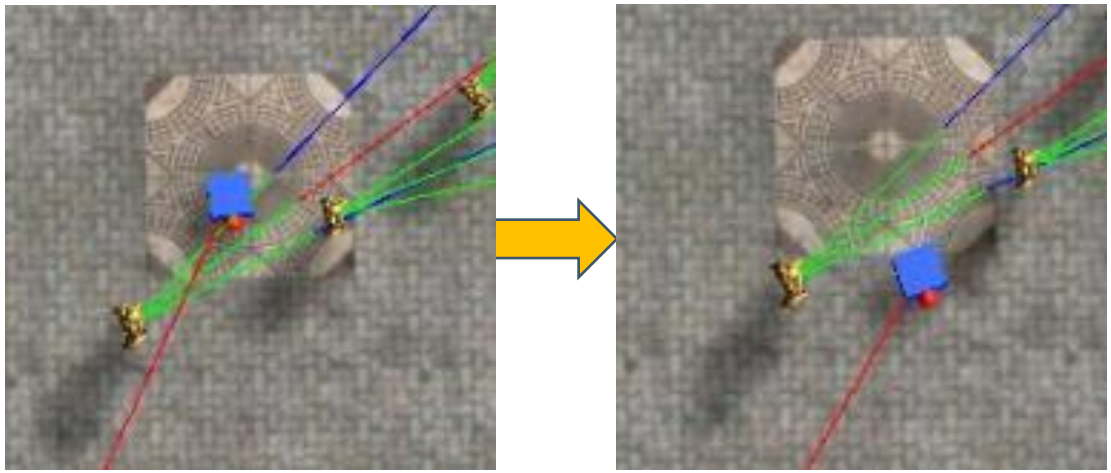
移動障害物回避の成功例

- 自律移動ロボットが静止して回避



静止して回避　　静止して回避

- 自律移動ロボットが旋回して回避

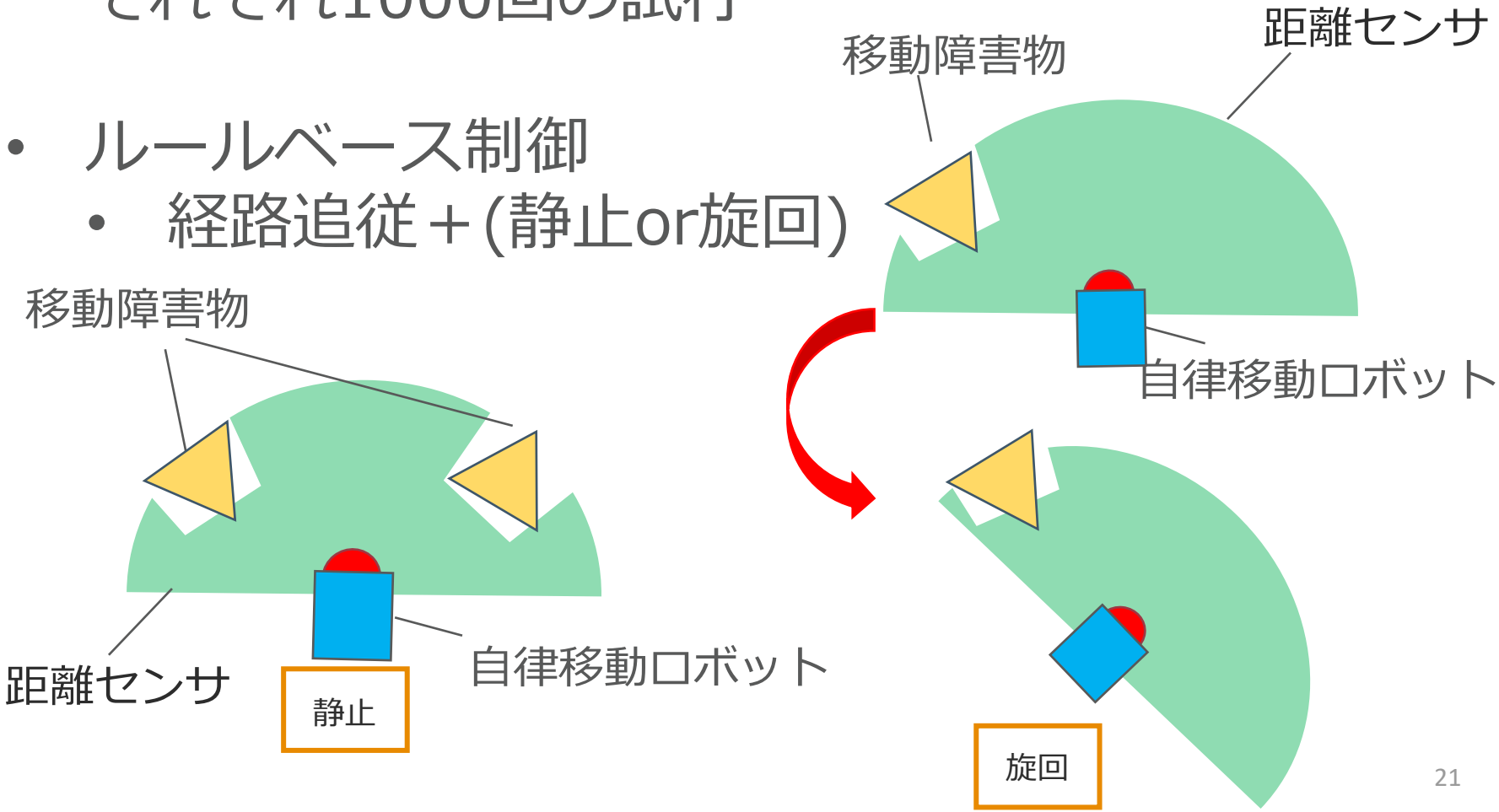


旋回して回避　　旋回して回避

精度検証(1/2)

- ルールベース制御モデルと学習モデルを用い
  それぞれ1000回の試行

- ルールベース制御
  - 経路追従＋(静止or旋回)



精度検証(2/2)

モデル	成功回数 (1000回試行)	ロボットが 静止中に衝突	ロボットが 移動中に衝突
ルールベース	783回	125回 (57.6%)	92回 (42.4%)
視線情報なしの学習モデル	873回	54回 (42.5%)	73回 (58.5%)
視線情報ありの学習モデル	**906回**	**42回 (44.6%)**	**52回 (55.4%)**

- 学習モデルはルールベース制御より高精度
- 視線情報ありのモデルの方が高精度
- 静止中のロボットに移動障害物が衝突する事例が失敗の4割以上

考察

- 学習モデルはルールベース制御より高精度
  - 状況に合わせた回避行動を学習
- 視線情報ありのモデルの方が高精度
  - 視線から外れるための移動を学習
- 静止中のロボットに移動障害物が衝突する事例が失敗の4割以上
  - ロボットに気づかせる工夫
  - 静止中のロボットと衝突しても安全な素材で構成

おわりに

- まとめ
  - 移動障害物回避のシミュレーション環境を作成
  - 強化学習を用いた移動障害物回避行動を獲得
  - 学習モデルを用いた回避行動を検証

- 今後の課題
  - 歩行者がロボットを認識しない状況の衝突回避
  - 搭乗者を考慮した障害物回避
  - 実環境での実証実験

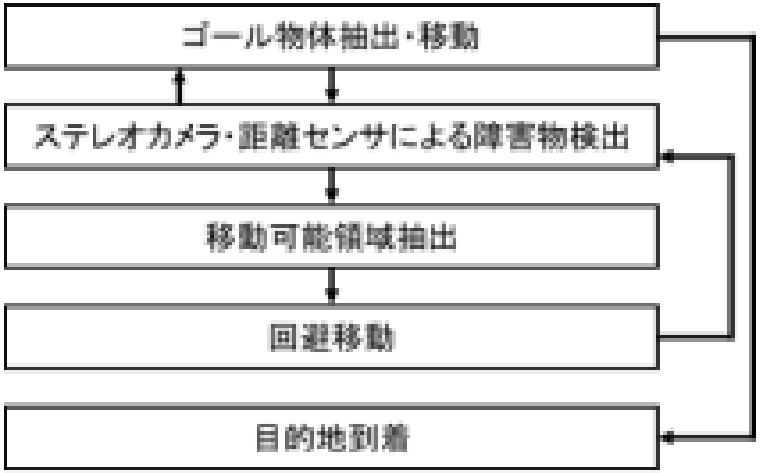
予備スライド

先行研究

- ステレオカメラと距離センサによる障害物回避

打井裕基一,芋野美紗子,土屋誠司, 渡部広一, “ステレオカメラと距離センサを用いた障害物検出による知能ロボットの自律移動手法,” 第14回情報科学技術フォーラム,vol.14, no.2, pp.291-292, 2015.

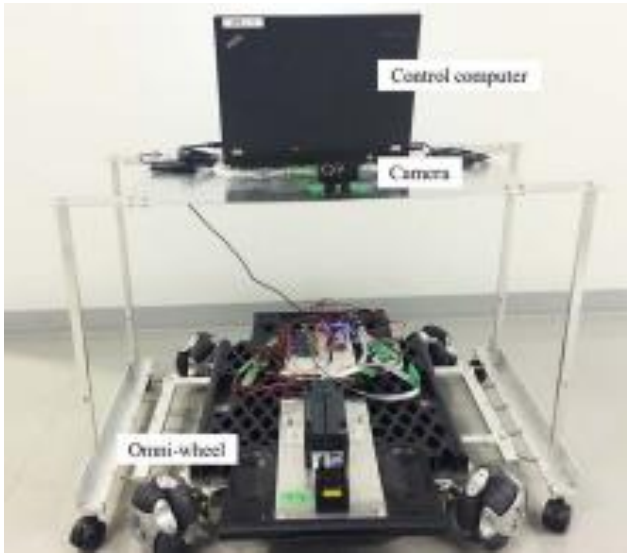
- 本研究との相違点
  - ルールベースによる障害物回避
  - 静止障害物の回避



先行研究

- 視線推定技術を使用して歩行者の視線に合わせた移動速度制御
  - Y. Sato and H. Igarashi, “Safe Driving Support of the Omni Directional Mobile Vehicle Using Gaze Measurement,” In Proceedings of the International Workshop on Nonlinear Circuits, Communications, and Signal Processing 2016.

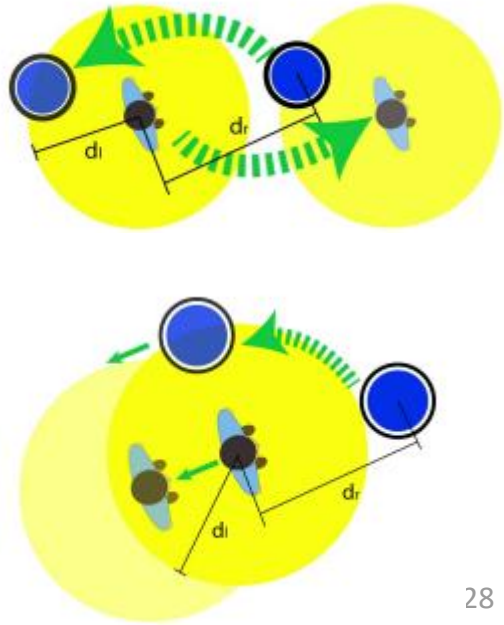
- 本研究との相違点
  - ルールベースによる制御
  - 障害物の回避を行わない



先行研究

- 歩行者の回避行動を定義し
  それぞれの行動に有効な経路計画を提案
  - 浅井悠佑, 廣井慧, 米澤拓郎, 河口信夫, 人の回避行動を考慮した移動ロボットの経路計画法の検討," マルチメディア，分散，協調とモバイルシンポジウム 2019.

- 本研究との相違点
  - ルールベースによる制御
  - 歩行者の行動が単一の環境で
    各実験を行っている



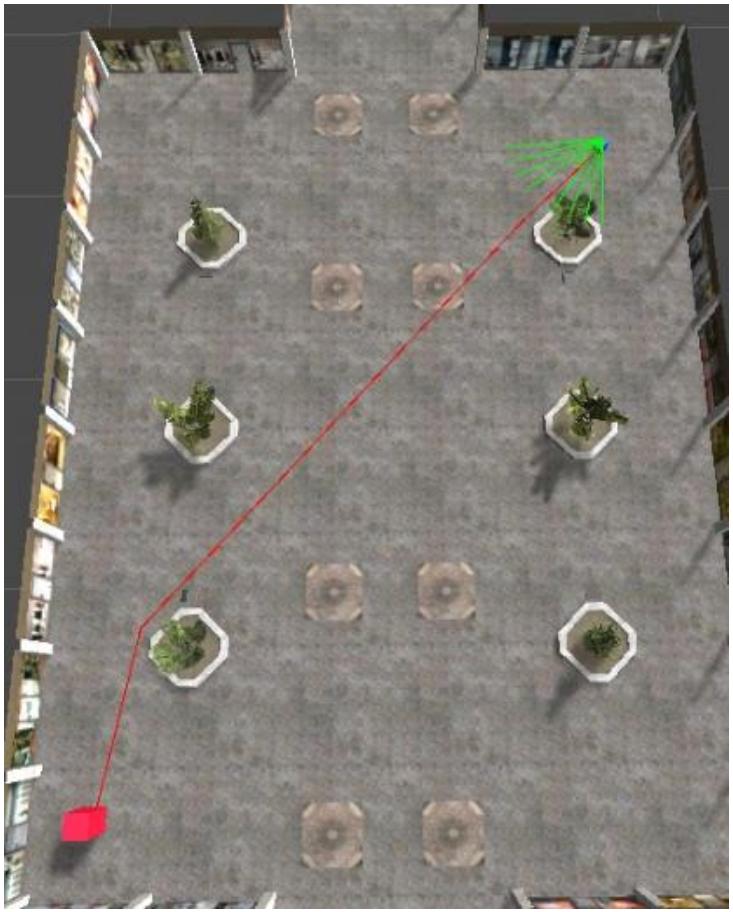
BellaBot

- RGBDカメラ・赤外線センサ・LiDARで環境認識
- AI音声
- 数10種類の表情

本体寸法	565×537×1290mm
ロボット重量	55kg
本体材質	ABS/アルミニウム合金
充電時間	4.5時間
バッテリー持続時間	12-24時間（交換式バッテリー）
安全性	速度:0.5-1.2m/秒（調整可能） / 登板角度：≤ 5°
積載量	最大40kg、10kg/トレー
位置決め方法	レーザーだけの高精度な位置決めが可能

目的地への走行

- Navmesh
  - ３D地形上でキャラクタの歩行可能な領域を設定
- NavmeshAgent
  - Navmesh上で出発地点から目的地点までの最短経路生成（赤線）
  - 複数の経由点を直線で生成
  - 経路探索はA*探索アルゴリズム



自律移動ロボットの経路生成の様子

強化学習

- Soft Actor-Critic (SAC)
  - 最大の報酬和を取得するように
    方策と行動価値関数を同時に学習
  - 過去の経験を含め、大量の経験を学習に使用
  - 方策にランダム性を追加し、多様な行動を決定
  - 複雑な環境下の学習に適する

学習アルゴリズム

- ML-AgentsのSoft Actor-Critic(SAC) [9]を使用

Parameter	Value
Number of hidden layer	2
Number of hidden layer nodes	256
Learning rate	0.0004
Learning rate schedule	constant
Replay buffer size	70,000
Batch size	500
Number of learning steps	3,000,000

[9] A. Juliani, V.-P. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Mattar, D. Lange, “Unity: A General Platform for Intelligent Agents,” CoRR, 2018.

SACの詳細

表 4.1: yaml ファイルの設定

trainer_type:	sac
hyperparameters:	
learning_rate:	0.0004
learning_rate_schedule:	constant
batch_size:	500
buffer_size:	70000
buffer_init_steps:	3500
tau:	0.005
steps_per_update:	12.0
save_replay_buffer:	False
init_entcoef:	0.1
reward_signal_steps_per_update:	12.0
network_settings:	
normalize:	True
hidden_units:	256
num_layers:	2
vis_encode_type:	simple
memory:	None
goal_conditioning_type:	hyper
deterministic:	False
reward_signals:	
extrinsic:	
gamma:	0.99
strength:	1.0
network_settings:	
normalize:	False
hidden_units:	128
num_layers:	2
vis_encode_type:	simple
memory:	None
goal_conditions_type:	hyper
deterministic:	False
init_path:	None
keep_checkpoints:	100
checkpoint_interval:	100000
max_steps:	13000000
time_horizon:	256
summary_freq:	10000
threaded:	True
self_play:	None
behavioral_cloning:	None

SACの設定

Batch_size
　勾配降下の更新1回に使用される経験（観察, 行動, 報酬）の数
Buffer_size
　モデルの更新を行う前に収集する必要がある経験（観察, 行動, 報酬）の最大数
Learning_rate
　学習率
Schedule
　学習率が時間による変化
Buffer_init_steps
　学習開始前に、何ステップ分のランダムな行動を経験バッファに埋めるか
Tau
　SACモデル更新中のターゲットの更新の大きさ
Steps_per_update
　ポリシーの更新に対するエージェントのステップの平均的比率
Init_entconf
　訓練開始時にエージェントがどの程度探索するか

表 4.1: yaml ファイルの設定

trainer_type:	sac
hyperparameters:	
learning_rate:	0.0004
learning_rate_schedule:	constant
batch_size:	500
buffer_size:	70000
buffer_init_steps:	3500
tau:	0.005
steps_per_update:	12.0
save_replay_buffer:	False
init_entcoef:	0.1
reward_signal_steps_per_update:	12.0
network_settings:	
normalize:	True
hidden_units:	256
num_layers:	2
vis_encode_type:	simple
memory:	None
goal_conditioning_type:	hyper
deterministic:	False
reward_signals:	
extrinsic:	
gamma:	0.99
strength:	1.0
network_settings:	
normalize:	False
hidden_units:	128
num_layers:	2
vis_encode_type:	simple
memory:	None
goal_conditions_type:	hyper
deterministic:	False
init_path:	None
keep_checkpoints:	100
checkpoint_interval:	100000
max_steps:	13000000
time_horizon:	256
summary_freq:	10000
threaded:	True
self_play:	None
behavioral_cloning:	None

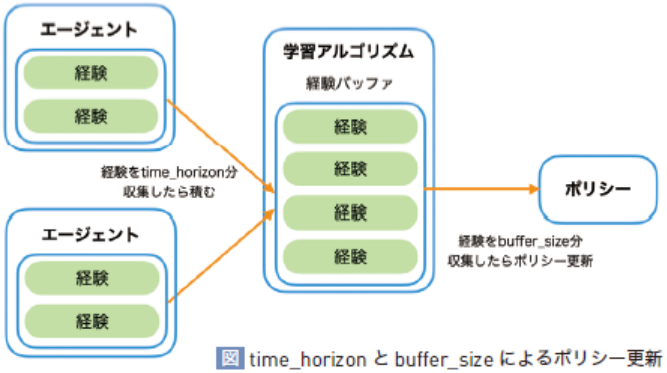
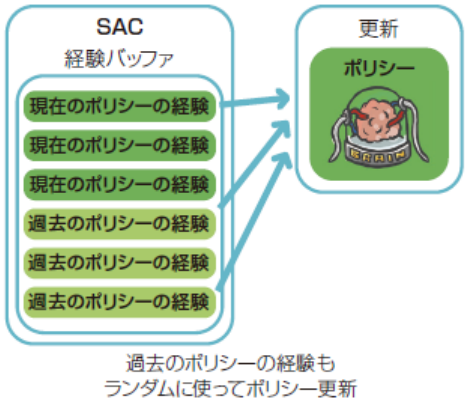
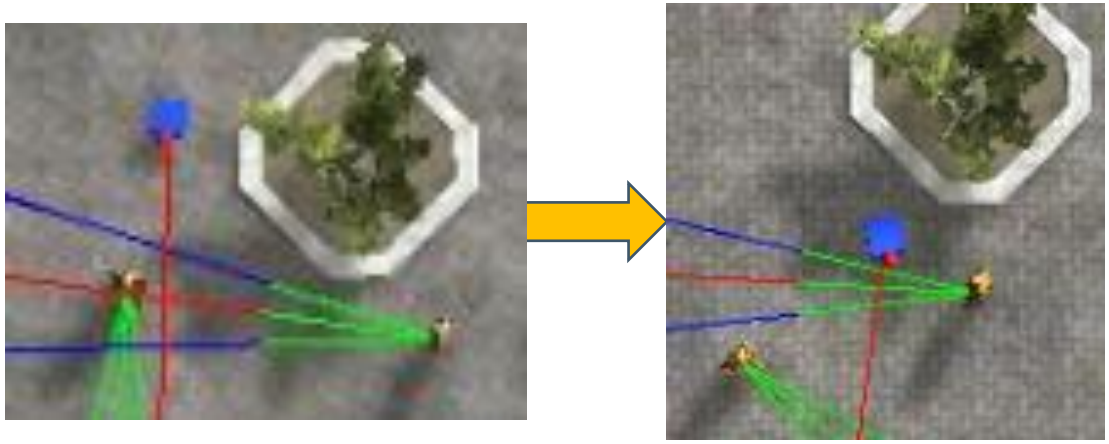


図 time_horizon と buffer_size によるポリシー更新

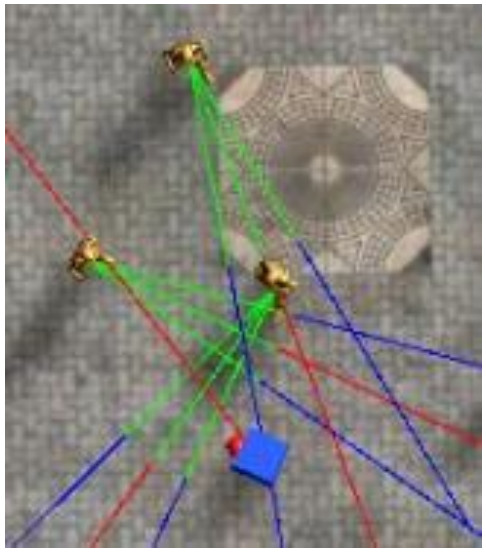


移動障害物回避の失敗例

- 衝突する原因
  - 静止障害物で検知の遅延
  - 回避した先に別の移動障害物が存在
  - 複数の移動障害物が密集



左の移動障害物を回避後、
右の移動障害物に接触の例



複数の移動障害物が密集した例

精度検証(学習環境)

モデル	成功回数	ロボットが静止中に衝突	ロボットが移動中に衝突
ルールベース	794回	116回 (56.3%)	90回 (43.7%)
視線情報なしの学習モデル	860回	80回 (57.1%)	60回 (42.9%)
視線情報ありの学習モデル	**907回**	**71回 (76.3%)**	**22回 (23.7%)**

学習結果

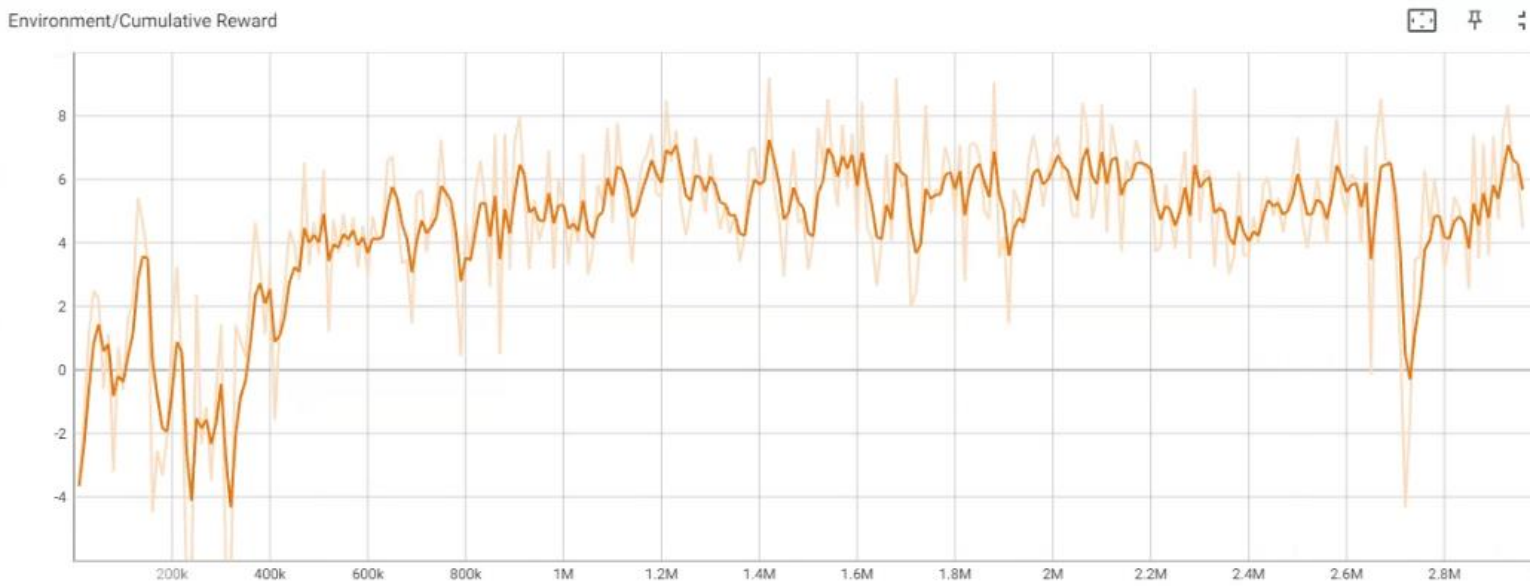
- 1万ステップ当たりの平均累積エピソード報酬
  - 50万ステップまで増加
  - 50万~300万ステップまで小さな増減

学習結果

- 1万ステップ当たりの平均累積エピソード報酬
  - 60万ステップまで増加
  - 60万~300万ステップまで小さな増減

移動障害物の視線情報あり（現在と１ステップ過去）モデル

平均累積エピソード報酬



ステップ数

学習結果

- 1万ステップ当たりの平均累積エピソード報酬
  - 50万ステップまで増加
  - 50万~300万ステップまで小さな増減

移動障害物の視線情報なしのモデル

平均累積
エピソード
報酬

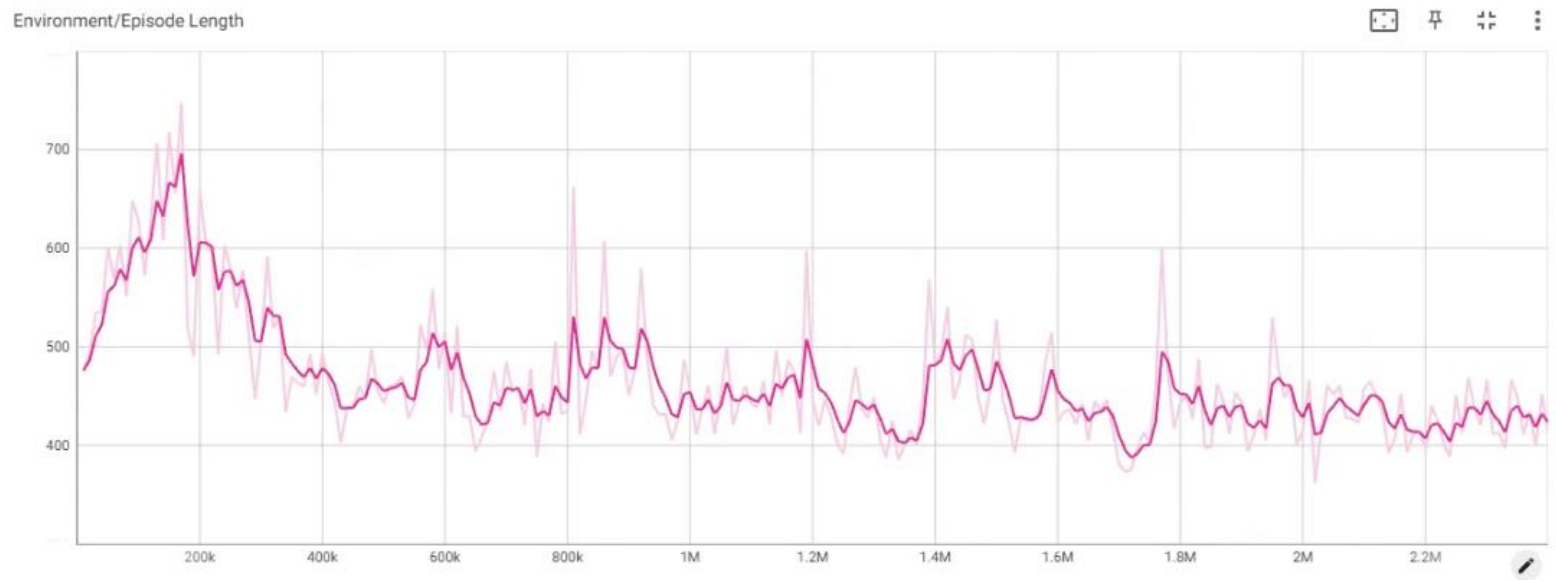
ステップ数

学習結果

- 1万ステップ当たりの平均累積エピソード報酬
  - 50万ステップまで増加
  - 50万~300万ステップまで小さな増減

移動障害物の視線情報なしのモデル

平均累積
エピソード
報酬



ステップ数

学習結果

- 1万ステップ当たりの平均累積エピソード報酬
  - 50万ステップまで増加
  - 50万~300万ステップまで小さな増減

移動障害物の視線情報なしのモデル

平均累積
エピソード
報酬

ステップ数

A* 探索

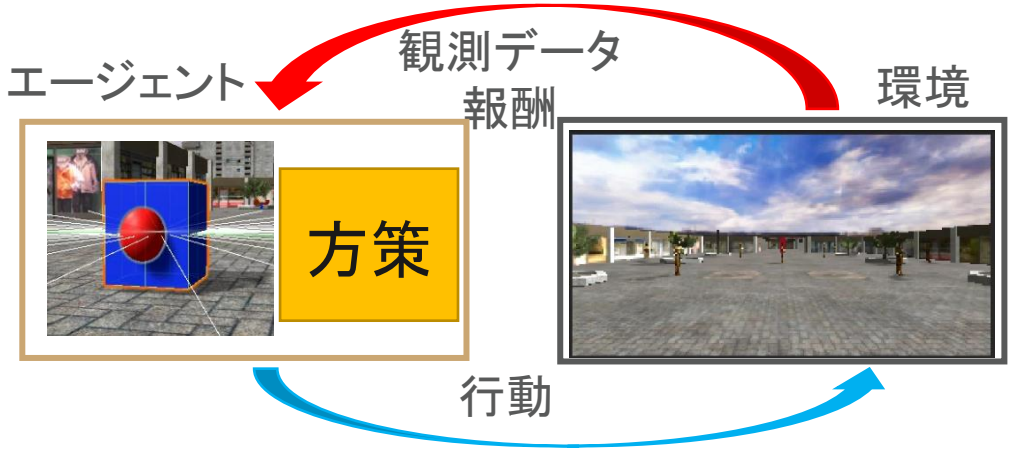
- スタートから現時点までのコスト
- 現時点からゴールまでの予想コスト
- コスト最小の経路を優先的に探索する方法

Navmeshの設定

視線検知の実装方法

- ➢ 自律移動ロボットに線が接触
  - ➢ 方角と本数の情報を取得
- ✓ 方角は計算軽量化のため
  30度ごとの配列
- ✓ 配列の値が本数

強化学習



- 観測データを使用して行動を決定し
  行動に応じた報酬を取得
- 方策
  - 観測データを入力し、行動を出力する関数
- 行動価値関数
  - 観測データと行動を入力し、報酬を出力する関数
- 経験
  - 観測データと行動と報酬の組み合わせ

経験に応じて報酬和を最大化するように方策を更新