Vol.2020-ICS-198 No.13
2020/3/10

RC カーを用いた自動運転車両シミュレーション環境の構築

織田 智矢[1,a)]　横山 想一郎[2]　山下 倫央[2]　蕨野 貴之[3]　大岸 智彦[3]　田中 英明[3]

概要：自動運転車の群制御を実現する上で，信号機の制御といった環境側から車両群の誘導・制御を行うアプローチだけでは十分ではないため，各車両の自律性を保った上で，交通流量を向上させる行動を獲得する手法を検討する必要がある．本研究では，自動運転車両同士（車両群）における協調行動の獲得の可能性と交通流量への影響を検証することを目的として，電動ラジオコントロールカー（RC カー）を用いたシミュレーション環境を構築する．

キーワード：自動運転，交通シミュレーション，強化学習

A Study on Self-Driving Vehicle Simulation Environment using RC Cars

1. はじめに

自動車は現代で生活する上では欠かせない存在になっており，近年では単体での自動運転車両の社会実装にむけて研究・開発が進んでいる．[1] しかし，便利な車社会が発達していった一方，様々な社会問題となっている．それらのうちの重大な社会問題として交通渋滞が挙げられる．日本国内における交通渋滞による損失例として，年間 12 兆円の経済的損失 [2]・救急車の遅れによる生存率の低下 [3]・移動時間の約 40%は渋滞によるもの [4] など様々挙げられる．また世界に目を向けると社会的損失は計り知れない．

重大な社会問題である渋滞を解決すべく，自動車の交通流量をシミュレートする研究は数多く取り組まれており [5] [6] [7]，様々な手法が存在する．しかしながら，その殆どがソフトウェアシミュレータを用いたものであり，実環境を走行する車の状態を十分表現できているとはいえない．そこで本研究では，交通渋滞を解消する様々な手法を検証．検討する目的で，ソフトウェアシミュレーション環境と，実際の車を使用する環境との中間に位置する，RC カーを用いたシミュレーション環境を構築した．また，交通渋滞は，様々なシチュエーションが存在するため，ルールベースでの最適化が難しい．そこで深層強化学習 [8] を用いて，交通流量の最適化を行う．

[1] 北海道大学 工学部
School of Engineering, Hokkaido University
[2] 北海道大学大学院情報科学研究院
Faculty of Information Science and Technology, Hokkaido University
[3] 株式会社 KDDI 総合研究所
KDDI Research, Inc.
[a)] oda@complex.ist.hokudai.ac.jp

2. 作成したシミュレーション環境の構成

本章では実験に使用したシミュレーション環境の構成について解説する．

図 1　シミュレーション環境の外観
Fig. 1　Appearance of the simulation environment

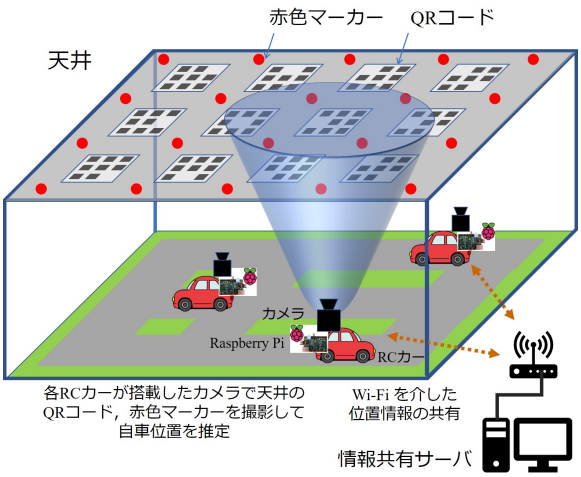

図 2　システムの構成図
Fig. 2　System configuration diagram

2.1　実験室の環境

実験に使用したコースは横 5m, 縦 8m であり，天井には自己位置推定用の QR コード・赤マーカ (図 3) を設置した．この天井のマーカまでの高さは 2.5m で，0.91m おきに赤マーカが設置されている．また格子状に設置された赤マーカの中心に QR コードを設置した．これは，QR コードのデコードには 1 秒近く掛かり，毎回これを処理していると RC カーの走行制御が出来なくなってしまうため，初期位置の決定には赤マーカと QR コードを用いて，それからの自己位置推定には赤マーカのみを使用することにした．

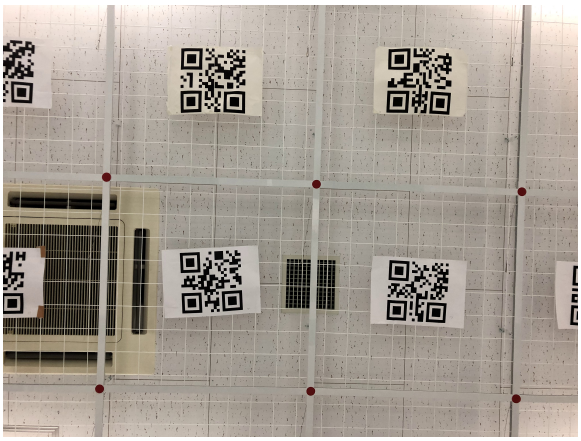

図 3　RC カーから撮影された天井に設置されている QR コードと赤マーカ
Fig. 3　QR code and red marker installed on the ceiling taken from the RC car

2.2　RC カーのハードウェア構成

本節では RC カー単体の構成に関して，ハードウェアの観点から解説する．本シミュレーション環境では，株式会社タミヤ製の 1/10 スケールの RC カー [9] を用いる．RC カーに，Raspberry Pi や大容量の社外製 LiPo バッテリーを搭載するため，サイズが比較的大きいこの車体を採用した．図 4 は，RC カーのカバーを外し内部を撮影したものである．Raspberry Pi は，レーザカッターで専用に切り出したアクリル板の台座にネジ止めされている．Raspberry Pi のモデルとしては，Raspberry Pi 3 Model B+ を採用している．

RC カーには Raspberry Pi Camera Module V2 カメラモジュールを搭載している．このカメラを RC カーの後部にマウントした．

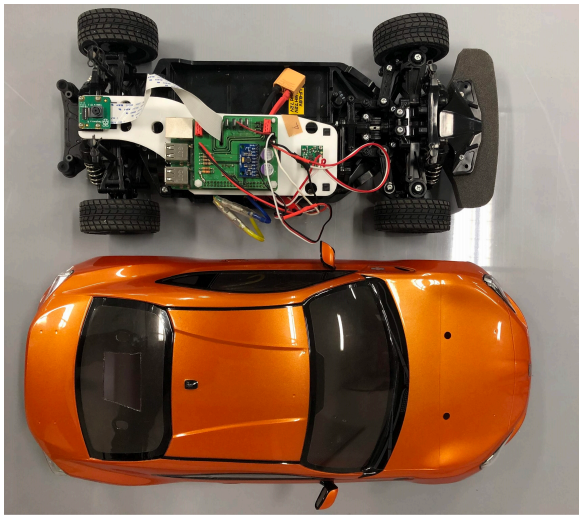

図 4　RC カー内部の外観
Fig. 4　Exterior of RC car

2.3　システムアーキテクチャ

本節では，RC カー単体の構成に関して，ソフトウェアの観点から説明する．図 5 は，各 RC カーのソフトウェアのシステム構成と処理の概要を示している．

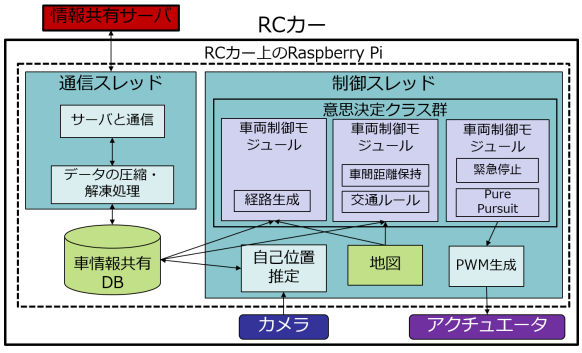

図 5　RC カーを用いたシミュレーション環要のアーキテクチャ
Fig. 5　Architecture of simulation environment using RC cars

2.3.1　自己位置推定

RC カーを用いたシミュレーション環境における車両の走行路はすべて同一の平面上に存在するため，天井方向から見ろしたときのコースの左上を原点とした，２次元平面

上でのメートル単位の座標により車両の位置を表す．また，車両の進行方向と y 軸方向のなす角度により車両の方向を表す以降，こうした座標系をワールド座標系と記述する．天井は走行路と平行な平面であるため，天井に存在する QR コードおよび赤色マーカーの位置および方向もまたワールド座標系で表すことができる．カメラは車両の中心に進行方向を下方として鉛直上向きに取り付けられており，撮影される画像は左上を原点とし，縦 w[px]，横 h[px] の範囲を持つ．また，カメラのレンズ歪みは無視できるものとする．こうしたカメラが撮影する画像に関する座標系を，以降ではカメラ座標系と記述する．このとき，車両の実空間上での座標を (x_c, y_c)[m]，方向を α として，QR コードおよび赤色マーカーを 2 次元の同次座標で表すと，ワールド座標系からカメラ座標系への変換行列は式 1 で表される．ただし，$T(x,y), S(x,y), R(\theta)$ はそれぞれ，x 軸方向，y 軸方向に関する平行移動，拡大・縮小，回転移動を表す座標変換行列とする．図 6 にワールド座標系とカメラ座標系の関係を図示する．

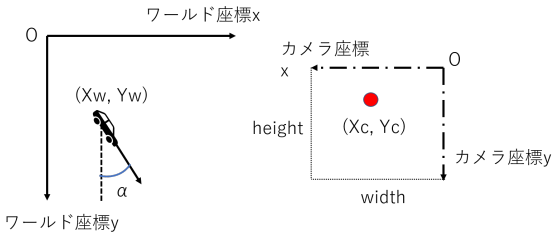

図 6 座標変換
Fig. 6 Coordinate transformation

$$A = T\left(\frac{w}{2}, \frac{h}{2}\right) S(f,f)\, R\left(-\frac{\pi}{2} - \alpha\right) T(-x_c, -y_c) \quad (1)$$

f[px/m] の具体的な値は，Raspberry Pi Camera Module V2 モジュールの画角は width 方向に 31.1°，height 方向に 24.4° であるため, 式 2 より 1m が何 px か計算することが可能である．

$$f = \frac{image\_height}{ceil\_height \times \tan(24.4) \times 2.0} \quad (2)$$

2.3.2 自己位置推定アルゴリズム

カメラ画像から読み取られた QR コード・赤マーカから，自己位置推定を行うアルゴリズムを説明する．車両の位置および方向が任意の値をとると仮定したとき，式 1 により，カメラ画像に映る QR コードおよび赤色マーカーの座標群を計算することができる．カメラ画像の認識結果との矛盾が最も小さくなるようなカメラの位置および方向を求めることで位置推定を行う．カメラの位置および方向を一意に決定するためには，読み取った QR コードまたは赤色マーカーの実空間上での取り付け位置を特定する必要がある．そのため，第 1 回目の位置推定において，OR コードおよび赤色マーカーの両方を認識し，QR コードとのカメラ画像中の位置関係から，各赤色マーカーの実空間上での取り付け位置を決定する．それ以降は，赤色マーカー群の座標のみを検出し，前ステップでの位置推定の結果を用いて，ステップ間のカメラ画像中での赤色マーカーの移動量が小さくなるような対応関係を仮定することで，赤色マーカーの実空間上での位置を決定する．検出された各赤色マーカーと実空間上での取り付け位置と対応関係が求められれば，2 つ以上の赤色マーカーが読み取られているときに，カメラの位置および角度の推定結果が一意に求められる．

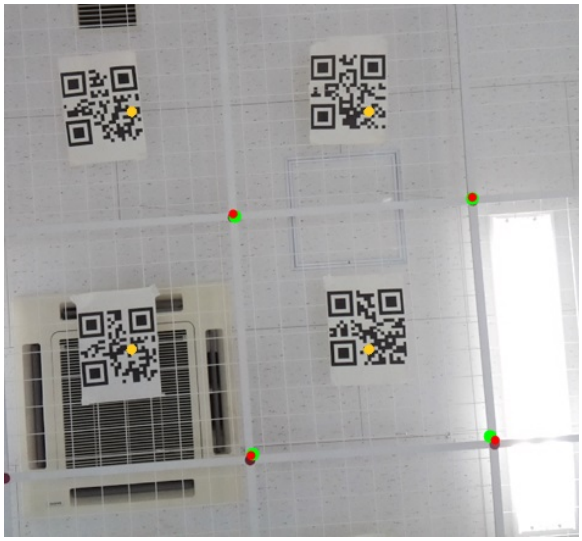

図 7 車両の位置推定で参照される座標群の例
Fig. 7 Example of coordinate group referenced in the RC car position estimation

車両の位置および方向は次の手順により推定される．

(1) カメラで画像を撮影し，カメラに映る QR コードのなかで最も中心に近いものについて，カメラ座標系上での位置・方向および ID を求める．
(2) QR コードの ID から，実空間上での QR コードの取り付け位置を求める．車両の位置を QR コードの直下と仮定し，実空間上での車両の方向を QR コードの読み取り結果から計算する．
(3) 実空間上に存在する赤色マーカー群の座標を，仮定された車両の位置および方向の値を用いて，式 1 からカメラ座標系上に変換する．カメラの撮影範囲内に存在する赤色マーカー群の座標集合を $A = \{a_1, a_2, ..., a_m\}$ とする．A の例を図 7 における黄色の点で示す．
(4) カメラ画像中に映る赤色マーカー群を求め，カメラ座標系での座標の集合を $B = \{b_1, b_2, ..., b_n\}$ とする．B

の例を図 7 における赤色の点で示す．

(5) B に含まれるすべての座標点について，A に含まれる座標点との一対一のペアを求める．

(6) 手順 5. により得られたペアのとおり A の座標群から B の座標群へ変換する行列を，回転および並進のみの制約のもとで最も 2 乗距離の誤差が小さくなるよう SVD により求める [10]．得られた変換行列を D とする．図 7 における緑色の点は A を D で変換することで得られる点の例である．

(7) D の表す回転および並進の値から，カメラの実空間上での位置および角度を次の通り求める．

- カメラ位置に関して，D が表す並進の値に対して $T(-p_{t-1})\,S(f,f)\,R(angle_t - \pi/2.0)$ で表される変換行列を左からかけることで得られる座標点とする．ただし，p_{t-1} は手順 3. で仮定した座標を表す．
- カメラの方向に関して，手順 3. で仮定した角度から変換行列に対応する角度を引いた値とする．

2.3.3 車両制御アーキテクチャ

RC カーを制御する上でそれぞれ意思決定を行うモジュール群をまとめたアーキテクチャを導入した．それぞれのモジュールの集合は，目標経路・目標速度・アクチュエータへの出力の 3 つから構成される．また，アクチュエータへの出力のモジュール集合には優先度付きのフローを導入した．図 8 はそれぞれのモジュールの階層とフローを示している．

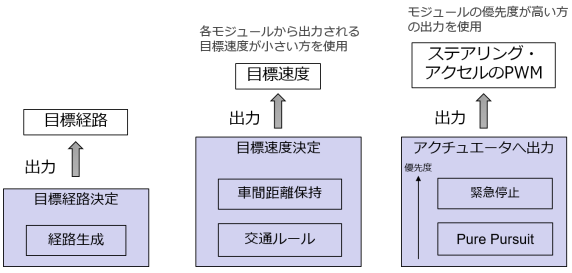

図 8 モジュール群の階層構造
Fig. 8 Module group hierarchy

2.3.4 地図の表現方法

本シミュレーション環境に用いたマップは図 9 に示されているように，12 個の交差点からなるものである．また，この地図を表現する方法として，有向グラフを用いた．

2.3.5 経路生成

RC カーはマップ上の目的地までの経路を動的に生成し，その経路上をライントレース方式に走行制御を行う．本節では，その具体的な経路生成について説明する．

(1) 現在の RC カーの前方かつ最も近い交差点インスタンスの頂点から，目的地の交差点 ID を含む交差点インスタンスの頂点までのグラフ上の道のりを，ダイクストラ法によって求める．

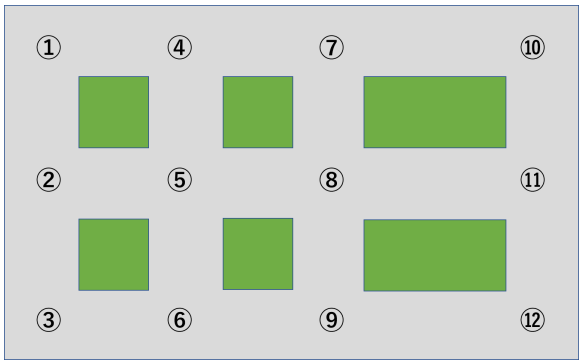

図 9 全体の地図
Fig. 9 The map of this environment

(2) 求められた経路に基づいて，交差点インスタンスの頂点が持つ座標情報から，3 次スプライン補完によって詳細な経路を生成する．

(3) 交差点インスタンス内の右折・左折・の経路に関しては，そのままスプライン補完によって生成すると，所望の経路が得られないため，交差点インスタンスが保持する制御点を経由するようにする．

スプライン補完に渡す交差点インスタンスの頂点の集合はそのまますべてを渡して計算するのではなく，交差点インスタンス間は直線になるように，交差点インスタンス内部は制御点を含めるように，一つの頂点ずつ生成して行き，最後のそれらの生成された経路をまとめる．

2.3.6 最適速度モデル (OVM)[11]

自車の前方に他車がおり，先行車とみなされる場合，最適速度モデルを用いて目標速度を求める．自車と先行車両との距離は直接的な距離ではなく，目標パス上の道のり D として定義される．

本来の最適速度モデル式は，加速度を決定するような式であるため，今回はこの式の速度を決定している部分のみを抜き出し使用した．

入力の D は $[0,3]$ になるように正規化し，その値に目標速度 $(target\_speed)$ をかけて決定した．車両の目標速度を決定する最適速度モデルの計算式は，

$$V_{target} = (tanh(D_normalize - c) + tanh(c)) \times target\_speed \tag{3}$$

と表される．

式 3 において，c は係数で， $c = 1.0$ と実験的に決定した値を設定する．

2.4 RC カーの制御

アクセルとステアリングについては，フィードフォワード制御を行う．これは，予め PWM パルスの出力と実測の値との対応を求めるために，キャリブレーションを行う．そして，実際に走行を行う時は，その対応表の最も目標に近い値の PWM パルスを出力する．これは，フィードバッ

ク制御と比較を行ったが，使用する RC カー機材のモータのトルクが大きく，あまり細かい制御ができず値が離散的になったため，PID 制御 [12] だと振動する現象が見られた．そのため，このようにフィードフォワード制御を導入した．このアクセルとステアリングの制御に関しては今後改善を行いたい．

2.4.1 ステアリング制御アルゴリズム

具体的なステアリング制御に関して，この節で解説を行う．目標の経路上を走行するように制御するために，Pure Pursuit アルゴリズム [13] を導入した．これは，自己位置と，目標の経路を与えることによって，目標のステアリング角度を算出するものである．

Pure Pursuit アルゴリズムとは，目標パス上の一定距離先の点を目標点 P_{target} とし，車両がその点に到達するように旋回制御を行う手法である．

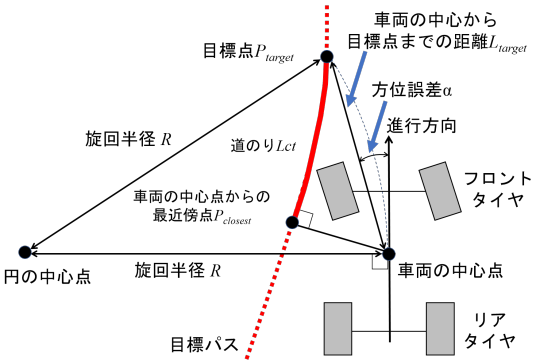

図 10 Pure Pursuit アルゴリズムの概要
Fig. 10 Pure Pursuit algorithm overview

図 10 に示されるように，車両の中心点から目標パス上の最近傍点を $P_{closest}$ とする．最近傍点 $P_{closest}$ から目標パス上を進行方向に道のり L_{ct} だけ進めた点を目標点 P_{target} とする．ここで，道のり L_{ct} は，現在の車両の速度を $V_{current}$ とすると，

$$L_{ct} = 0.1 \times V_{current} + 0.4[m] \tag{4}$$

と表される．パス上の目標点 P_{target} までの長さを L_{target}，現在の車両の向きと P_{target} までの向きの方位誤差 α を用いると，目標の旋回半径 R は

$$R = \frac{L_{target}}{2\sin(\alpha)} \tag{5}$$

と表される．

次にこの旋回半径 R になるように目標のステアリング角度を算出する．

目標のステアリング角度 s は

$$s = \arctan\left(\frac{2W\sin(\alpha)}{L}\right) \tag{6}$$

と求められる．

式 6 から算出された目標のステアリング角度 s を用いてステアリングの制御を行う．

2.5 情報共有サーバ

シミュレーション環境において複数車両を走行させる場合，衝突を回避するために他車との位置情報の共有が不可欠である．RC カーは Wi-Fi のソケット通信を用いて，情報共有サーバとの通信を行う．通信プロトコルは速度を重視するため，UDP を使用する．また，位置情報や交差点の情報だけではなく，パケットの順序や正当性を保証するために，タイムスタンプを付加する．

各 RC カーは表 1 に示される自車の位置情報を情報共有サーバに送り，情報共有サーバは，表 1 に示されているデータを全 RC カーから集約して，他の RC カーへとブロードキャストする．このブロードキャストされるデータは全台分の情報を含んでいる．

表 1 共有される各車両の位置情報
Table 1 Shared information for each RC car

項目	範囲	データ型
車両 ID	[0,31]	整数値
x 座標 [m]	[-1.0,6.0]	実数値
y 座標 [m]	[-1.0,12.0]	実数値
進行方向 [rad]	[-π,π]	実数値
速度	[0, 1.0]	実数値
タイムスタンプ	[0, 1.797693e+308]	実数値
前の交差点 ID	[0,11]	整数値
現在の交差点 ID	[0,11]	整数値
次の交差点 ID	[0,11]	整数値
優先度	[-1,5]	整数値
前の交差点からの距離	[0,50]	整数値
次の交差点への距離	[0,50]	整数値
ストップ指示フラグ	[0,1]	論理型
緊急停止使用フラグ	[0,1]	論理型

2.6 ソフトウェアシミュレーション環境の構成

基本的には RC カーを用いたシミュレーション環境のモジュールを再利用し，両シミュレーション環境が独立しないように作成する．また，強化学習が行いやすいように同期式のシミュレータを作成する．

2.6.1 システムアーキテクチャ

システムのアーキテクチャを図 11 に示す．

具体的な実現方法としては，RC カーを用いたシミュレーション環境のときの Raspberry Pi 上で動作していたプログラムをクラスとして定義し，同期式で動作させるためすべて単一のスレッドに変更する．次に，自己位置推定のアルゴリズムは不要であるため削除し，アクチュエータへ

PWM を出力していたモジュールを運動方程式に変更する．そして，運動方程式から計算される次の自己位置の値を車情報共有 DB へと送信する．

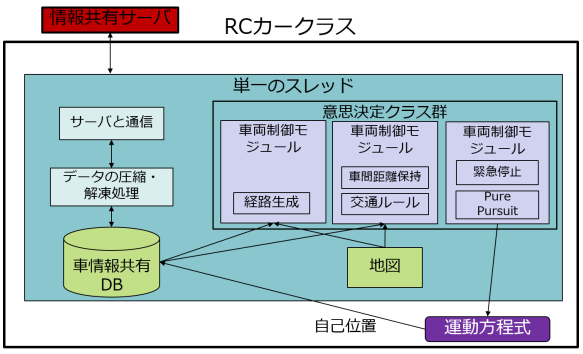

図 11　ソフトウェアシミュレーションのアーキテクチャ
Fig. 11　Software simulation architecture

3. RC カーを用いたシミュレーション環境の性能評価

3.1 自己位置推定誤差

自己位置推定が正しく動作しているか実験を行った．車両の角度を一定方向に固定し，取得する画像全体に赤色マーカと QR コードが存在する (2.0,2.0)$[m]$ の 2 つの地点を開始点として y 方向または x 方向に 0.15m ずつ 7 回ほど車両を移動しそれぞれの地点で位置を推定した．

位置推定を行ったそれぞれの点（真値）と推定値とのグラフを図 12 に示す．なお，y 軸方向に車両を移動して測定した結果については，グラフの x 軸と y 軸を反転させている．

また，RC カーに取り付けられたカメラは垂直上方向を向いていないため，そのまま自己位置推定を行うと一定方向にズレが生じる．そこで，真値と比較して，カメラの傾きを比較し，キャリブレーションを行った．これにより，自己位置推定の誤差がキャリブレーション前にすべての点の誤差平均が 0.084m だったのが，0.028m まで小さくできた．

平均誤差が偏った 0.03m 以下であれば，車両のふらつきによる衝突等もないと考えられるため，十分な精度の位置測位を行えたと考える．

3.2 走行制御誤差

Pure Pursuit アルゴリズムを用いたステアリング制御の目標パスの追従精度を検証する．図 9 に示されるコースの 6 番と 7 番を通る周回コースを作成し，RC カーを 3 分間走行させる実験を行った．実験においては車両の目標速度は一定に設定した．図 13 は RC カーが 3 分間周回コースを走行時の目標パス（赤線）と RC カーの走行軌跡（青線）を示している．RC カーは 3 分間でこのコースを 7 周

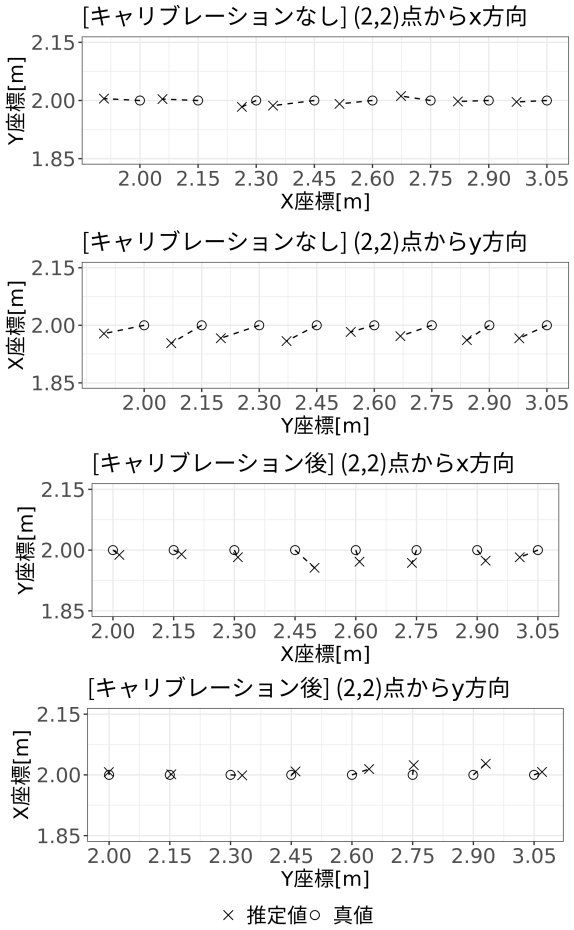

図 12　自己位置推定の誤差
Fig. 12　Error in self-location estimation

した．目標パスに対してカーブで外側にふくらむが，直線部分ではふらつかない．周回ごとの走行軌跡の差も見られず，毎周ほぼ同じ経路を走行していることが確認できる．RC カーの中心座標と目標パス上の最近傍点の距離を誤差として，3 分間の毎ステップの誤差を平均値を算出して，目標パスの追従精度を評価する．この実験において，3 分間の RC カーの走行における平均誤差を，表 2 に示す．RC カーの幅が 0.14m，コースの交差点内の車線の幅が 0.25m であるため，目標ラインから離れることなく，対向車線の車両とおおよそ衝突せずに走行できる範囲に含まれている．そのため，Pure Pursuit アルゴリズムを用いたステアリング制御は RC カーを用いたシミュレーション環境において有用であることが確認できた．

表 2　走行実験結果
Table 2　Running experiment results

ID	car1	car2	car3	car4	car5	car6
XTE[*1] $[m]$	0.061	0.052	0.067	0.058	0.055	0.066

[*1] XTE (cross track error) 車両の現在位置から目標経路に引いた垂直距離の誤差

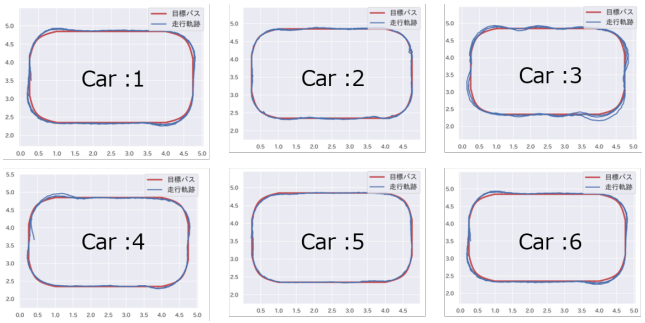

図 13　6 台の走行軌跡
Fig. 13　Trajectory of six RC cars

4. ソフトウェアシミュレーション環境上で強化学習・性能評価

4.1 交差点内での強化学習の適用

車全体の交通流量を増加させるために，交差点内での待ち時間を最初化するような行動を獲得させるために実験を行う．交差点内部での待ち時間を最小化させるためには，交通ルールに則った走行だけでなく，時としては自身より優先度の低い車を先に行かせたほうが，全体として流量が増加し，待ち時間が減る場合がある．今回はそういったシチュエーションが発生するようにシナリオを固定し強化学習を行った．

4.1.1 行動空間

交差点進入時での判断のみで良いため，シンプルに “停止”，“交通ルールに従って走行” の 2 つのみとした．

4.1.2 観測空間

今回は数多くのシチュエーションや設定を検証したかったため，交差点内部で行動判断に有効と思われる特徴量を選んで，入力を行った．表 3 に，観測するベクトルの内容を示した．4 方向からの進入予定の車の数はそのまま数値で入っているが，あとの値は one-hot となっている．ぜんぶで長さ 18 の一次元ベクトルであり，これをそのままニューラルネットの入力とする．ネットーワークの構成や学習設定は後述する．

表 3　観測ベクトル
Table 3　Observation space

特徴量	値	例
4 方向からの進入予定の車の数	int	[1,2,0,0]
自分の進路	one-hot	[1,0,0]
交差点ルール上進入可能か	bool	[0]
自分か先頭車両か	bool	[1]
他の交差点先頭車両の進路	one-hot	[1,0,0,0,1,0,0,0,0,0,0,0]

4.1.3 エピソード設定

エピソードは交差点に進入する一定ステップ前に開始し，交差点を通り過ぎる・一定ステップの経過によって終了する．エピソード終了後は，エピソード開始条件を満たすまでシミュレータを動かし続け，その条件を満たした時に次のエピソードとして入力を受け付ける．エピソード開始条件とは，強化学習制御対象車が交差点に近づいていてかつ，交差点の他の車が存在することである．

4.1.4 報酬

今回の目的は一時的な交差点内での優先順位の変更によって，全体の交通流量を増加させることを目的としているため，短期的な即時報酬と，長期的な報酬として，最終ステップに付与する 2 種類の報酬を設定した．以下に具体的に報酬の設定を説明する．また，今回は負の報酬を使用しない設定とした．

- 交差点内で停止行動を選んだときに，交差点内を通過した車がいたら+1
- エピソードが終了した後，一定ステップ経過後までに交差点を通過したらそのステップ時点で+1 とし，最終的に計算された割引報酬を最後のステップに付与

1 つめの報酬の意図としては，停止行動をおこなったときに経過するステップ数に対する報酬の割引を補填するために設定した．2 つめの報酬の意図としては，全体の交通流量を増加させることが目的であるため，交差点内の流量に比例するような設定にした．

4.1.5 ネットーワーク構成

本実験の強化学習では PPO をベースに構成されているため，方策のネットワークと価値のネットワークの 2 種類存在する．価値の値はそのまま出力したが，方策はその後 Softmax 関数を通してそれぞれを行動の確率とし，その確率に従って行動する．また，その他パラメータは表 4 に示す通りである．

表 4　ハイパーパラメータ
Table 4　Hyper parameters

パラメータ	値
エピソード内の最大ステップ数	100
観測するステップ数	80
エピソード終了後から交差点を観測するステップ数	1000
割引率 (gamma)	0.99
Horizon	300
Lambda	1
Clip_param	0.2
バッチサイズ	64
方策学習率	1e-4
価値学習率	3e-4

4.2 学習結果

1000 エピソード学習を行ったあとの報酬の変化を図 14 に示す．この図から見て取れる通り，分散は大きいものの，

初期状態よりは報酬が得られるようになっている．

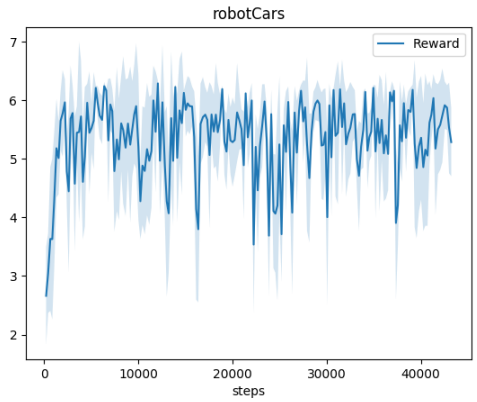

図 14 強化学習中の報酬の遷移
Fig. 14 Reward transition

また，表 5 に示すように学習後のモデルを使用して 20 分間の走行実験を行った．これにより，強化学習車を一台導入するだけで，全体の交通流量が増加したと言える．

表 5 20 分間の走行実験結果
Table 5 20 minute running experiment result

実験設定	通常の交通ルール	強化学習後
総移動距離 [m]	4367.61	4727.18
平均総移動距離 [m]	623.95	675.31

5. 結論

本研究では，自動運転車両同士の交通流量を増加させるための RC カーを用いたシミュレーション環境を作成し，その環境で通常の交通ルールに則った走行実験を行い，ルールベースでの安定した RC カーの走行が実現できた．また，強化学習のためのソフトウェアシミュレーション環境を作成し，その上で交差点内の交通流量の最適化の強化学習を行った結果，同じステップ数でより多くの距離を走行していることが確認できた．つまり，強化学習によって制御された車が 1 台導入されることによって，全体の交通流量が増加したと言える．しかしながら，通常の交差点ルールを使用した走行と大きな差は生まれなかった．これは，学習・テストを行う上で所望のエピソードを多く観測出来なかったためであると考えられる．

参考文献

[1] Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., Jesus, L. F. R., Berriel, R. F., Paixão, T. M., Mutz, F. W., Oliveira-Santos, T. and de Souza, A. F.: Self-Driving Cars: A Survey, *CoRR*, Vol. abs/1901.04407 (online), available from ⟨http://arxiv.org/abs/1901.04407⟩ (2019).

[2] : 国土交通省　業績計画書/達成度報告書，https://www.mlit.go.jp/road/ir/ir-perform/ir-perform.html.

[3] : 厚生労働省資料，https://www.mhlw.go.jp/stf/shingi/2r98520000025aq3-att/2r98520000025ax5.pdf.

[4] : 国土交通省　交通流対策資料，https://www.meti.go.jp/committee/sankoushin/sangyougijutsu/chikyu_kankyo/yakusoku_souan_wg/pdf/005_07_00.pdf.

[5] Liang, X., Du, X., Wang, G. and Han, Z.: Deep Reinforcement Learning for Traffic Light Control in Vehicular Networks (2018).

[6] van Arem, B., van Driel, C. J. G. and Visser, R.: The Impact of Cooperative Adaptive Cruise Control on Traffic-Flow Characteristics, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 7, No. 4, pp. 429–436 (online), DOI: 10.1109/TITS.2006.884615 (2006).

[7] Drew, D. R.: *TRAFFIC FLOW THEORY AND CONTROL* (1968).

[8] : Python で学ぶ強化学習: 入門から実践まで，講談社 (2019).

[9] : 電動 RC カーシリーズ No.530 トヨタ 86 (TA06 シャーシ)，https://www.tamiya.com/japan/products/58530/index.html.

[10] 玉木徹: 姿勢推定と回転行列，電子情報通信学会スマートインフォメディアシステム研究会（SIS），pp. 59–64 (2009).

[11] Bando, M., Hasebe, K., Nakayama, A., Shibata, A. and Sugiyama, Y.: Structure stability of congestion in traffic dynamics, *Japan Journal of Industrial and Applied Mathematics*, Vol. 11, No. 2, pp. 203–223 (online), DOI: 10.1007/bf03167222 (1994).

[12] 日高浩一: 「制御工学―技術者のための，理論・設計から実装まで」（豊橋技術科学大学・高等専門学校制御工学教育連携プロジェクト　著），計測と制御，Vol. 53, No. 2, pp. 155–155（オンライン），DOI: 10.11499/sicejl.53.155 (2014).

[13] Paden, B., Čáp, M., Yong, S. Z., Yershov, D. and Frazzoli, E.: A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles, *IEEE Transactions on Intelligent Vehicles*, Vol. 1, No. 1, pp. 33–55 (online), DOI: 10.1109/TIV.2016.2578706 (2016).