コンピュータ理工学特別研究報告書

題目

レーザスキャナを用いた模型自動車の制御に関する研究

学生証番号　1544755

氏名　白石 風太

提出日　平成 31 年 1 月 22 日

指導教員　蚊野 浩

京都産業大学
コンピュータ理工学部

要約

自動運転技術は，周囲の環境を把握するセンサを活用して，自動車を制御する必要がある．そのための車載センサとしてカメラ，レーザスキャナ(LIDAR)，ミリ波レーダーなどが利用される．本研究では単一の LIDAR を用いて周囲の環境を入力し，それを Raspberry Pi で解析し，解析データから模型自動車を自律走行させるプログラムを開発した．そのために，LIDAR センサから取得したデータをグリッドマップに変換し，障害物の位置を判定した．

グリッドマップは空間を格子状に区切った地図のことである．グリッド（格子点）ごとに，障害物の領域／障害物のない領域／障害物の背後の領域，の三つのいずれかに判別する．今回は，模型自動車を中心に 5000㎜×5000㎜ の空間を 1 グリッド 25㎜ の精度で認識できるように設定した．LIDAR センサから計測データを取得するたびにグリッドマップを更新し，その時点での模型自動車の目標点を決定した．従って，模型自動車はあらかじめ決めたグローバルな目標点に向かうのではなく，周囲環境に関する局所的な情報に基づいて自律走行を行なった．目標点は模型自動車を中心にした判定円上の前方領域にあるとした．判定円の半径を 500㎜ から 1500㎜ に設定し，最も制御性に優れた半径を求める実験を行なった．判定円から目標点を決めるアルゴリズムは以下のようにした．

① 判定円上で，模型自動車が通過できる幅を持つ区間を抽出する．
② 抽出した区間の中から，直進方向に近い区間を選択する．
③ 選択した区間の中心部分に向けた経路が走行可能であれば目標点とする．

目標点が決まれば，その目標点に向かうためのステアリング角を算出し，走行させた．実験の結果、判定円の半径が 1000㎜ の時に，比較的スムーズは動きで障害物を避けながら走行させることができた．

目次

1章 序論

自動運転技術は各国で研究・開発が進められている．自動車関連規格の標準化活動を行っている米国SAE(Society of Automotive Engineers)は自動運転技術を6つのレベルに分け，評価している．そのレベルのうち，3以上にあたるものはシステムが主体であり，現在の自動運転技術が目指すべきレベルである．多くの自動車メーカーがレベル3，4に到達する自動運転技術の開発に力を入れている．例えば，トヨタ社は，高速道路に限定して，2020年までにレベル3の自動運転を行える自動車の開発を目標にしている．日本政府も，自動運転技術に関するロードマップとして「官民 ITS 構想・ロードマップ 2018」[2]を策定している．この文書の中では，自動運転のための要素技術や利用形態，産業への影響，法整備などの指針が示されている．

自動運転技術のレベル向上にはカメラやミリ波レーダー，LIDAR(Light Detection and Ranging)といった周囲環境を把握できるセンサが重要となる．これらのセンサはそれぞれに長所短所があり，実際の自動運転車においては複数のセンサを組み合わせ，それぞれのセンサが持つ短所を補うように制御される．矢野経済研究所の予測[1]によると，自動運転技術を支えるセンサの市場規模は，2020年に1兆6,688億1,000万円，2030年に3兆2,755億2,700万円に達すると予測されている．このように，車載用のセンサは自動運転に必須のデバイスであり，市場も大きいため注目を集めている．

車載センサ自体は，自動車の周囲に関する空間的なデータを計測するデバイスである．その空間データから自動車の運転に必要な情報を抽出し，アクセル・ハンドル・ブレーキの制御を行う．センサから得た空間データを処理することや運転に必要な情報を抽出すること，および自動車を制御することは，自動運転技術の中枢であり高度なアルゴリズムが求められる．

車載センサの中で，LIDARは障害物までの距離を，信頼性良く高精度で測定することができる．実車を使った研究でも主要な車載センサとして利用されている．本研究では，掃除ロボットの制御用に開発された小型のLIDARを用いて模型自動車を自律走行させた．本研究の目的は，組み込みシステムの技術や自動車を制御するための考え方などを習得することである．また，本研究の目標は，自動車周囲の局所的な空間情報を使って自律走行ができるシステムを制作することである．

2 章 自動運転技術の現状

2.1 実用化のレベル

表 2.1 に自動運転技術に関する SAE のレベルを示す．現在，国内ではレベル 2 までが実用化されている（例えば，2016 年に日産がセレナに搭載した自動車専用道路の単一車線を自動運転できるシステム）．レベル 3 に関して，アウディ社は 2017 年のアウディ A8 に条件を満たす自動運転が可能な自動車を開発し販売を行なっている(但し，日本では法律上販売できない)．

世界最大の家電見本市である CES 2019 において，トヨタ社が Lexus LS をベースとした自動運転の実験車を公開した．この自動車は，同社が開発を進めている高度安全運転支援システム(ガーディアン)を導入した車両である．ガーディアンとは人を主体としたシステムであり，事故が起こりそうな状況が差し迫っている場合に，ドライバによる操作とシステムを協調させ正確な回避を可能にするシステムである．また，ZF 社（ドイツの自動車部品メーカー）は自動運転を行う AI 対応のスーパーコンピュータ「ZF ProAI RoboThink」を発表している．このように自動運転に関する技術の開発や実験が続々と進められている．日本の自動運転技術の今後に関して，「ITS 構想・ロードマップ 2018」によると，2020 年に，移動サービス(タクシーやバス)の無人運転を限定地域で実施，物流サービス(トラック)の無人隊列走行を高速道路で実現，自家用車では自動運転レベル 3 を可能とした自動車の市場化を目指している．

表 2.1 自動運転技術のレベル

SAEレベル	概要
レベル0	運転の自動化はない
レベル1	速度かハンドルの制御をシステムが行う
レベル2	速度かつハンドルの制御をシステムが行う
レベル3	緊急時を除いた制御をシステムが行う
レベル4	限定エリアにおいて完全自動運転
レベル5	どの環境においても完全自動運転

2.2 車載センサ

自動運転に用いる車載センサは，人間の目に代わって，周囲環境の空間データを数値として計測する．現在のセンサ技術では，一つのセンサだけで周囲環境を十分に認識し自動運転させることは難しい．主要な車載センサとしてカメラ，LIDAR，ミリ波レ

ーダーの三つがあり，これらを組み合わせて利用する．それぞれのセンサの特徴を表2.2に示す．

カメラは主に車体の前方を観察するために用いられる．撮影した映像を人工知能(AI)によって解析し，障害物の位置や種類を認識することが可能である．カラーカメラを用いる場合，夜間や雨天など物体を観察しにくい状況では認識精度が悪い．

ミリ波レーダーは波長の短い電波を照射することで，センサの前方領域にある物体との距離を計測できる．ミリ波レーダーは他のセンサに比べると空間分解能において劣るが，ミリ波を放射しそれを検出するため，夜間や天候に左右されず計測できる．

LIDARは赤外線レーザを照射し，その光が反射するまでの時間から物体との距離を計測するセンサである．赤外線レーザはミリ波レーダーよりも波長が短い電磁波になるので空間分解能が高く，ダンボールや発泡スチロールのようにミリ波の反射率が低い物体にも対応できる．しかし，光線を用いて計測を行うので，その光線が遮断されてしまうと計測ができない．

表2.2 車載センサの特徴

センサ	長所	短所
カメラ	空間分解能が高く，道路標識や信号，車両・人などの障害物，道路面の白線などを認識させることができる．	夜間や雨・雪などの悪天候では，性能が悪い．
ミリ波レーダー	夜間や雨・雪でも十分に動作する．	電波の反射率が低い物体の検出ができない．
LIDAR	空間分解能はカメラとミリ波レーダーの中間程度．夜間や雨・雪でも利用可能．	豪雨，豪雪，霧といった天候が特に悪い条件では性能が悪い．

3 章 模型自動車と LIDAR を用いたシステムの構成

3.1 LIDAR センサとシステム構成

本研究では，自作の組み込みシステムを用いて市販の模型自動車を制御した．組み込みコンピュータとして，PIC マイコンと Raspberry Pi を組み合わせたものを使用した．この組み込みシステムと模型自動車は，2017 年度の特別研究 II[3]で使用したものと同様である．

センサとして図 3.1 の LIDAR センサ（Slamtec 社の RPLIDAR A2）を使用し，模型自動車の制御を行う．この LIDAR は，近赤外のレーザスポット光を周囲 360° 方向に時計回りに照射し，レーザ光が反射する時間から物体までの距離を求める．生データとして障害物の方向を示す角度と距離を取得できる．図 3.2 に LIDAR が計測する角度と模型自動車の位置関係を示す．LIDAR の 0° 方向が自動車の真後ろになるようにセットし，LIDAR の角度はそこから時計回りに進む．センサの性能を表 3.1 に示す．1° 程度の角度分解能で 10m ぐらいまでを計測することができる．

図 3.1 本研究で用いる LIDAR センサ（Slamtec 社の RPLIDA A2）

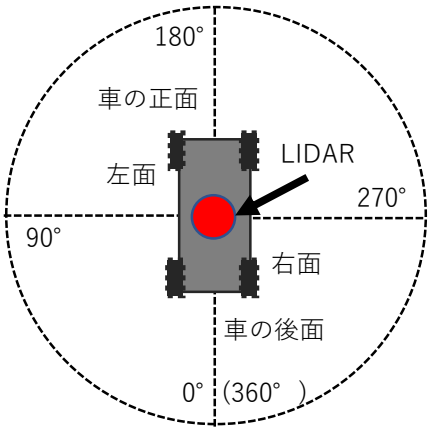


図 3.2 LIDAR の測定角度と模型自動車の位置関係

表 3.1 RPLIDAR の性能

項目	数値	備考
距離	0.15m～12m	
角度	0° ～360°	
距離分解能	0.5mm	
角度分解能	0.45° ～1.35°	回転の速さにより，約0.9°
計測時間	0.125msec	回転の速さによる
一秒間のサンプル数	8000個	回転の速さによる
電圧	5V	

図 3.3 にシステム全体のブロック図と模型自動車の外観を示す．LIDAR と Raspberry Pi は USB で接続する．Raspberry Pi から模型自動車を制御する部分は従来と同じで，Raspberry Pi から I2C インタフェースを介して PIC マイコンに指令を送り，PIC マイコンが模型自動車への駆動信号を発生する．今回開発するプログラムは Raspberry Pi で動作する．開発するためのターミナルとして WiFi で Mac Book を無線接続した．モバイルバッテリ２台搭載しており，一つを Raspberry Pi の電源に，他方を LIDAR の電源に用いた．

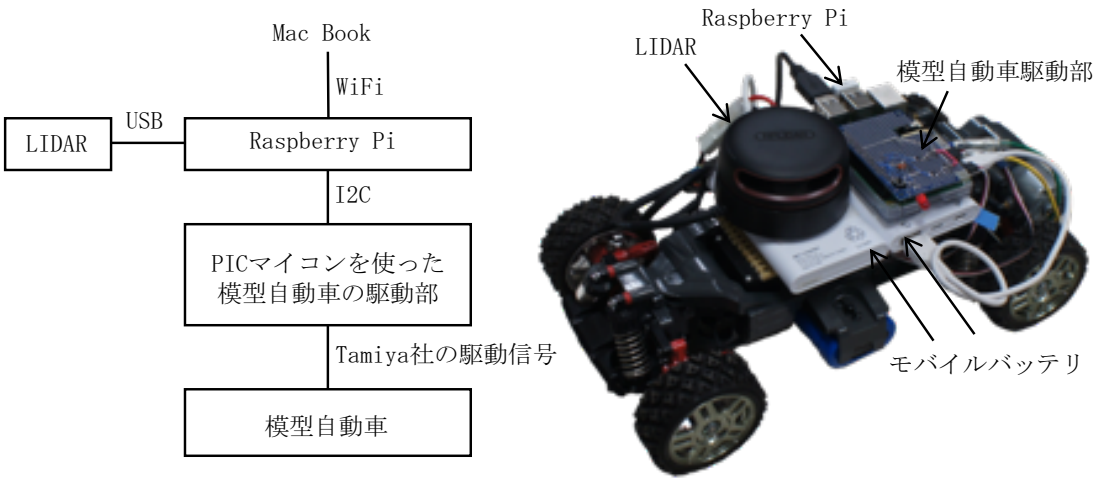


図 3.3 システム全体のブロック図と模型自動車の外観

開発するプログラムの概略フローチャートを図 3.4 に示す．まず，LIDAR で 360° 全周方向の距離データを入力する．生データを適切に処理した後，障害物の判断や経路の決定を行う，必要なステアリング角度と前進・後退の強さを PIC マイコンへの制御

コマンドとして与える．Raspberry Pi と Mac book の SSH 通信の接続方法について付録 1 に示す．

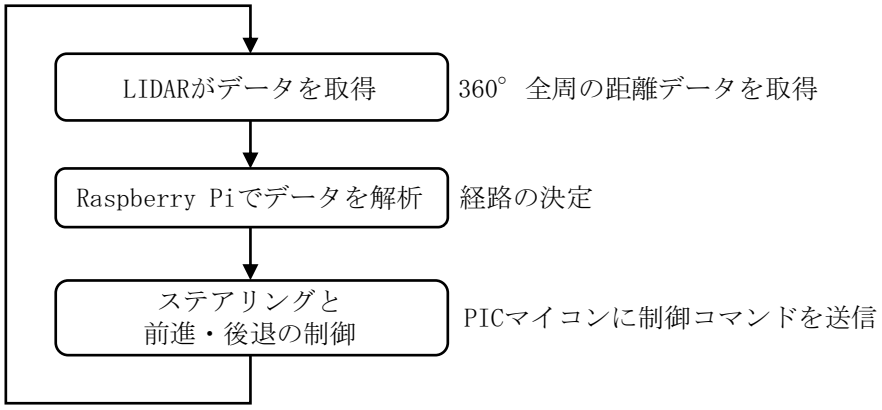


図 3.4 開発するプログラムの概略フローチャート

3.2 グリッドマップの作成

LIDAR の計測データを使った模型自動車の制御では，障害物を回避できる方向に車を導く必要がある．これを実現するために，環境を把握しやすいグリッドマップ(格子地図)を作成することにした．その理由を以下に示す．

(1) グリッドマップについて調査したところ,比較的容易に実現できることがわかった[4].

(2) グリッドマップを使うと走行可能な範囲を容易に把握できる.

(3) LIDAR は光線を照射し，照射方向の直線上のどこに障害物が存在するかを調べる．光線が反射する位置までには障害物がなく，その先は状態が不明な領域と言える．この特性が,グリッドマップの構造と類似しており,LIDARセンサとの相性が良い.

これらの理由からグリッドマップを採用した.

グリッドマップは，空間を正方形(グリッド)のマス目に分解して表現した地図(図 3.5)のことである．例えば図 3.5 は，200×200 の 40,000 個のグリッドで構成されており，一つのグリッドを 25mm×25mm とし，LIDAR を角度分解能 2° で測定した生データから生成したものである．図 3.5 では，中央がセンサの位置となっており，そこを中心として放射線状に赤・黒・緑の三つの領域に分割している．この三つの色は赤色が障害物，黒色が走行可能領域，緑色が不明領域である．今回作成するグリッドマップは，LIDAR で計測した角度ごとに三つの領域に分割する．障害物より手前側は走行可能領域，障害物の位置は走行不可能領域，障害物より後方は不明領域とする．グリッドの大きさを小さくすればグリッド数が多くなり，空間分解能が高くなる．また，LIDAR の角度分解能を小さくすれば細かな障害物にも対応することができる.

今回，5000mm×5000mm の空間を，グリッド幅 25mm，LIDAR の角度空間分解能 1° としてグリッドマップを作成することにした(図 3.6)．この場合，模型自動車の前方・後方・右側・左側それぞれに対して 2500mm の範囲までの空間を把握できる．

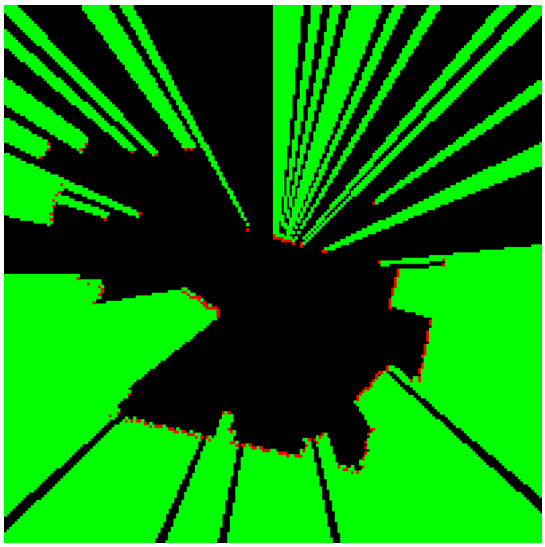


図 3.5 グリッドマップの例

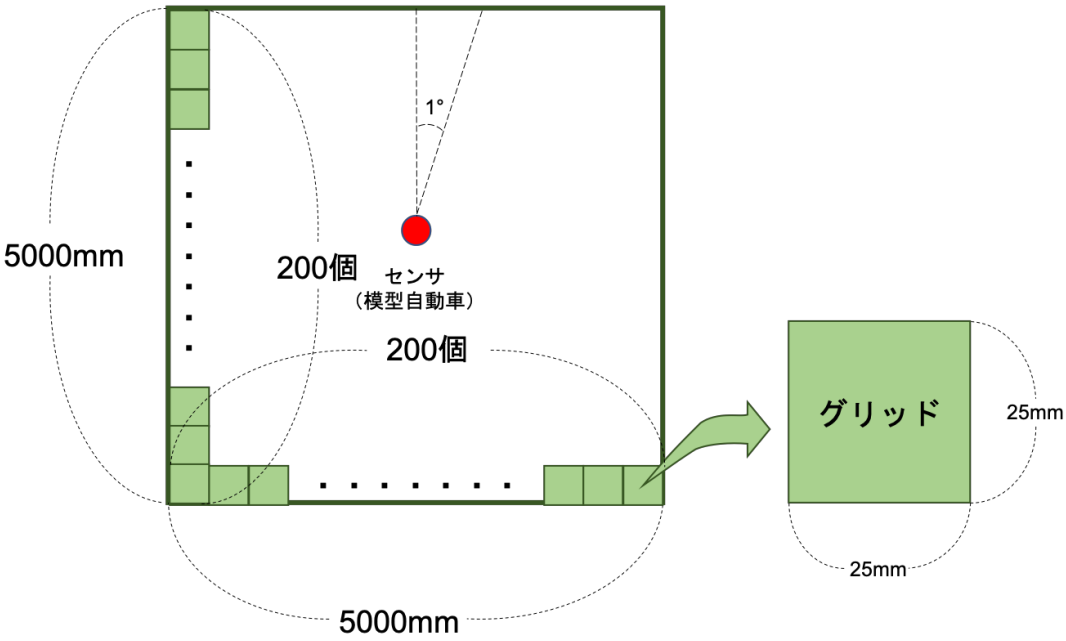


図 3.6 使用するグリッドマップの設定

3.3 グリッドマップを使った目標点の推定

今回の研究では，LIDAR で計測した生データからグリッドマップを生成する．そして，グリッドマップを生成するたびに目標点を設定し直し，自動走行させる方法を検討した．従って，この自動走行には通常の意味での目標点というものはなく，障害物に関する局所的な情報だけで自動走行させた．

グリッドマップから前方に障害物が無く直進可能であることがわかれば，正面を目標の方向とする．そうでない場合，障害物を回避するように適切な目標点を決めなければならない．その方法を，図 3.7 を用いて説明する．図の中央はグリッドマップを生成したときの模型自動車の位置を示す．5000㎜×2500㎜ の領域はグリッドマップの上半分である．今回は，模型自動車を前方に進行させる場合だけを検討する．青色の点線は模型自動車のステアリング角を一つの値に固定した場合の経路であり，円弧になる．この円弧はグリッドマップ中央の垂線に接する．判定円は，目標点を決めるための目安とする半円である．今回は判定円上の一点に目標点を設定する．判定円の半径を 500㎜ から 1500㎜ に設定して実験を進めた．

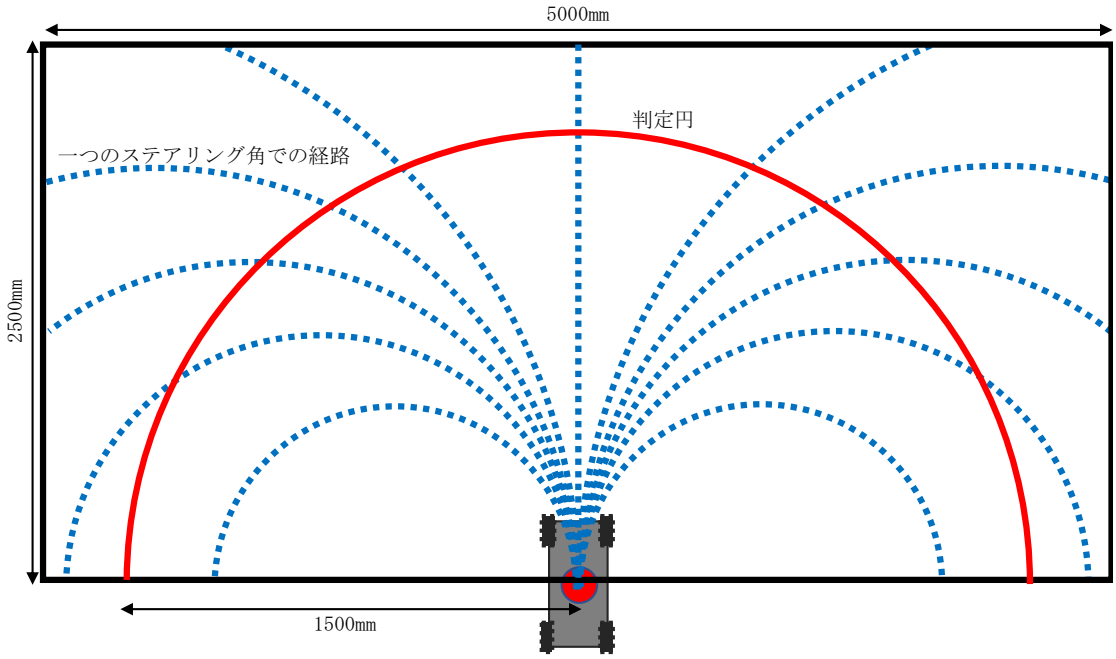


図 3.7 グリッドマップから中間目標点を決めるための説明図

このような条件において，次のアルゴリズムで判定円上に目標点を定めた．

① 判定円上のグリッドマップの値から，模型自動車が十分に通過できる幅を持つ区間を抽出する．

② そのような区間の中で，直進方向に最も近い区間を選択する．

③ 選択した区間の中央に向かうまでの経路が走行可能であればそこを目標点とする．ここで，「通過できる／できない」の判定において，模型自動車が図 3.7 で示すように，湾曲して走行することに注意しなければならない．このアルゴリズムは，可能な限り直進に近い走行をするように制御している．その理由は，ステアリング角を大きくすると湾曲が大きくなるため，判定円の位置では通過できると判定される場合であっても，そこに行くまでの経路で通過できない可能性が高まり，制御が複雑になるからである．

3.4 ステアリング角の算出

3.3 節のアルゴリズムで決定した目標点に向かうためのステアリング角（タイヤの切れ角）を決定しなければならない．模型自動車に設定するステアリングコマンドの数値は-15〜15 の整数になっており，その数値によって前輪のタイヤの切れ角が決まる．それにしたがって，実際に車が動く回転円の半径が決まる．今回，ステアリングコマンドの数値とそれに対応する回転円の半径を実測した．また，計測値と理論値を比較し，実験の妥当性を検証した．ここで，理論値[5]は式(1.1)から求める．式(1.1)において，R は回転半径、L は軸距(ホイールベース)，Tf は舵取り車輪の軸距 α は外側車輪の舵取り角度，β は内側車輪の舵取り角度である．今回の場合，L は 223㎜，Tf は 140㎜ であった．

$$R = \frac{\frac{L}{sin\alpha} + \sqrt{L^2 + (\frac{L}{tan\beta} + Tf)^2}}{2} \quad \cdots (1.1)$$

コマンドの数値ごとの内輪と外輪のタイヤ切れ角（実測値），回転円半径の理論値と実測値を表 3.1 に示す．この表から実測値が理論値とよく合っていることがわかる．

今回の研究では，コマンドの数値（-15 から 15）に対応する回転円の半径として，実測値を使用した．しかし，表 3.1 において，コマンドの数値 5 から-5 に対しては，半径が大きくなったため，実測せず，理論値から推定した値を用いた．

模型自動車の位置を原点として，目標点の座標(x,y)に到達するための回転円の半径を r とすると式(1.2)が成り立つ．式(1.2)から回転円の半径 r は式(1.3)になる．

$$y^2 + (x - r)^2 = r^2 \quad \cdots (1.2)$$

$$r = \frac{x^2 + y^2}{2x} \quad \cdots (1.3)$$

このようにして求めた回転円の半径を表 3.1 に当てはめ，最も近い回転円半径に対応するステアリングコマンドの数値を決定した．

表 3.1 ステアリングコマンドの数値とタイヤの切り角・回転円半径の関係

(a)左折する場合

ステアリングコマンド	タイヤ切り角(外輪(α))(単位：°)	タイヤ切り角(内輪(β))(単位:°)	回転円半径(理論値)(単位: mm)	回転円半径(実測値)(単位: mm)
15	16.5	21.0	769.9	762.5
14	15.2	19.0	835.5	815.0
13	13.9	16.9	915.1	867.5
12	13.0	15.4	984.6	928.8
11	12.0	14.0	1067.0	990.0
10	11.1	12.5	1166.5	1113.8
9	10.1	11.0	1289.0	1237.5
8	9.6	9.5	1416.8	1390.0
7	9.0	8.0	1583.3	1542.5
6	7.7	6.9	1834.1	1840.0
5	6.4	5.7	2185.4	2137.5
4	5.1	4.6	2712.6	2850.0
3	3.9	3.4	3591.8	3562.5
2	2.6	2.3	5350.9	5326.7
1	1.3	1.1	10629.5	10605.4

(b)右折する場合

ステアリングコマンド	タイヤ切り角(外輪(α))(単位：°)	タイヤ切り角(内輪(β))(単位:°)	回転円半径(理論値)(単位: mm)	回転円半径(実測値)(単位: mm)
-15.0	14.5	23.0	796.2	810.0
-14.0	14.0	20.8	841.9	868.8
-13.0	13.5	18.5	896.0	927.5
-12.0	12.5	16.8	969.5	1046.3
-11.0	11.5	15.0	1058.0	1165.0
-10.0	10.5	13.3	1166.7	1288.8
-9.0	9.5	11.5	1303.6	1412.5
-8.0	8.4	9.9	1488.7	1646.3
-7.0	7.2	8.2	1740.7	1880.0
-6.0	5.7	6.9	2130.2	2372.5
-5.0	4.1	5.6	2771.8	2865.0
-4.0	3.3	4.5	3446.0	3547.8
-3.0	2.5	3.4	4570.0	4671.8
-2.0	1.6	2.2	6818.6	6920.4
-1.0	0.8	1.1	13565.4	13667.2

4 章 目標点への制御性の検証と自動走行実験

4.1 基本的な制御性の確認

3 章で求めた目標点に正しく制御可能であること検証するために，図 4.1 のような状況を作り走行実験をした．この実験では，目標点を決定する処理を，走行開始時に一回だけ行い，その目標点に到達することを確認した．また，目標点の判定を行う円の半径を 1500㎜ にした．

図 4.1(a)は直進方向にのみ進路がある場合である．図 4.1(b)は直進方向に進路があり，それとは別な方向にも進路がある場合である．図 4.1(c)，(d)は左あるいは右に進路がある場合である．図 4.1(e)は複数の走行可能領域があり，それらが左右に別れているような場合である．図 4.1(f)は走行可能領域がない場合である．

図 4.1(a)の場合，正しく直進した．図 4.1(b)の場合も，直進方向を選択し，正しく直進した．図 4.1(c)，(d)の場合も，正しく左右に走行した．図 4.1(e)の場合，左右に別れている走行可能領域の中で，直進方向に近いものを選択して走行した．図 4.1(f)の場合，あらかじめ決めておいた距離で停止した．この実験によって模型自動車の基本的な制御性が確認できた．

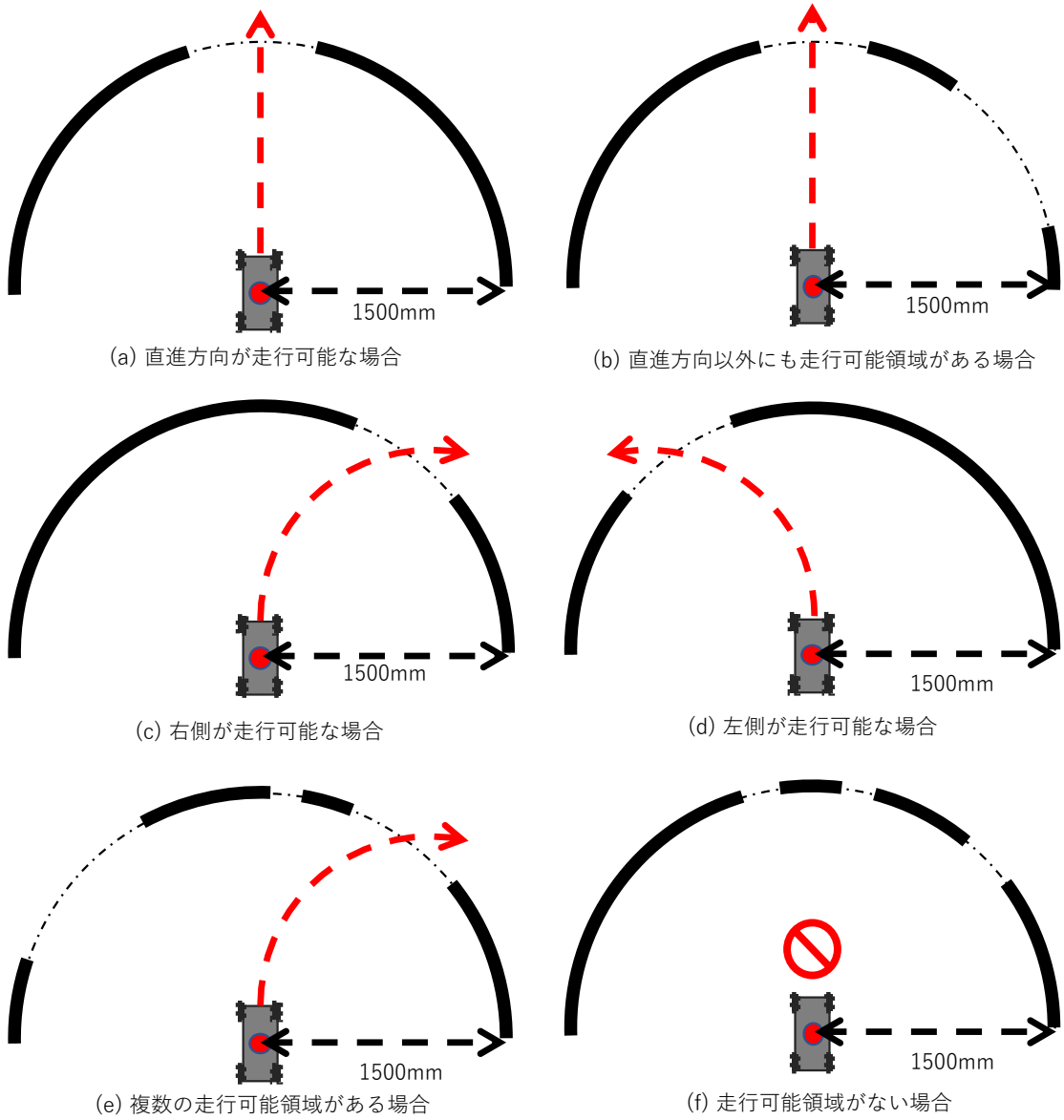


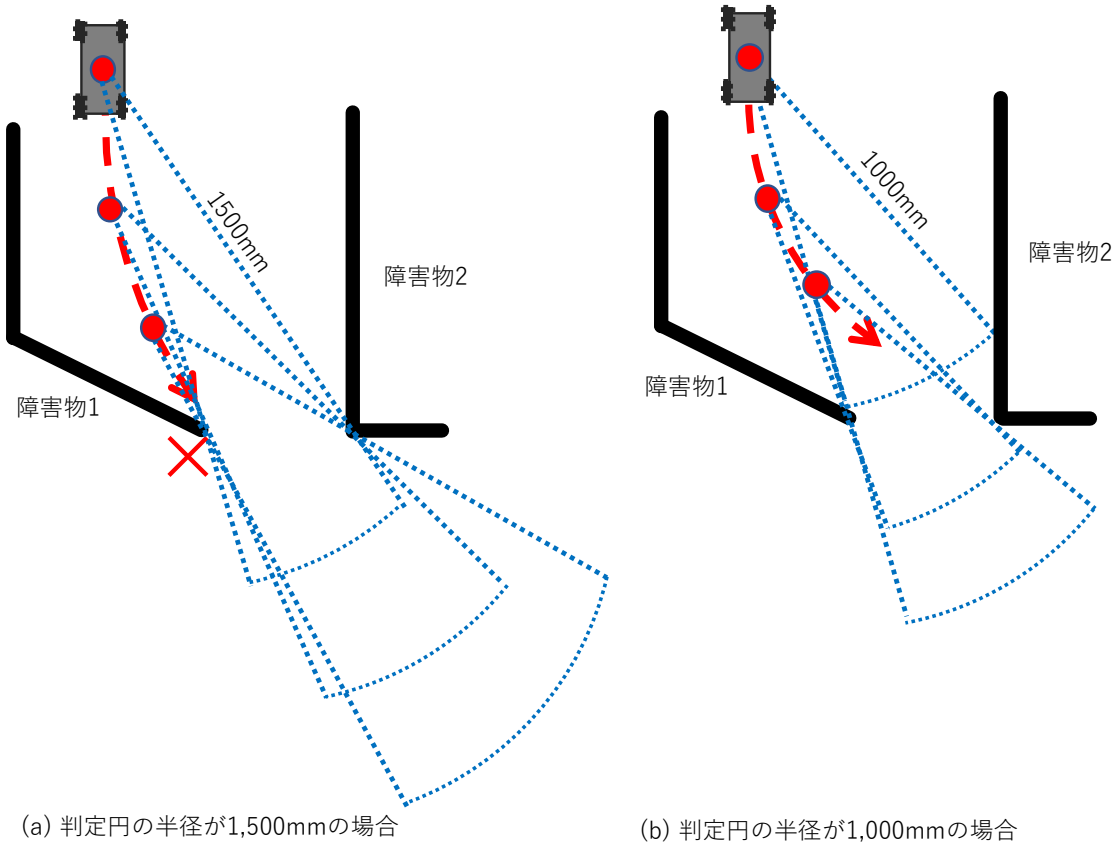
図 4.1 模型自動車の制御可能性を確認するための走行パターン

4.2 連続的な処理における制御性の確認

実際の走行では，周囲環境を測定するごとに目標点を再設定する．このように連続的な処理を行った場合の走行実験を行なった．図 4.2 に，その時の状況の一例を示す．

図 4.2 において，(a)は判定円の半径を 1500㎜ にした場合の走行経路，(a)は 1000㎜ にした場合の走行経路の模式図である．図 4.2(a)では，模型自動車は障害物 1 に近づきすぎストップしてしまった．図 4.2(b)では，障害物 1 と障害物 2 の隙間を通過することができた．図 4.2(a)で目標点に到達できなかった理由は，スタート時の目

標点が遠い場所にあるため回転円の半径が大きくなり，目標点に至るまでの経路で障害物に近い領域に入ってしまったためである．図 4.2(b)では，(a)に比べて早いタイミングで障害物 1 から遠ざかるように制御され，結果的に 2 つの障害物の隙間を通り抜けることができた．この実験から，今回の模型自動車の場合，判定円の半径は 1500mm では大きすぎ，1000mm の方が制御性がよかった．



(a) 判定円の半径が1,500mmの場合　　(b) 判定円の半径が1,000mmの場合

図 4.2 目標点の設定と制御を連続的に行なった場合の例

4.3 周回コースを用いた走行検証

周回路にいくつかの障害物を置いたテストコースを作った．コースの写真を図 4.3 に示す．コース内に大小いくつかの障害物を様々な位置に置き，模型自動車の動作を観察した．

図 4.3 テスト用の周回コース

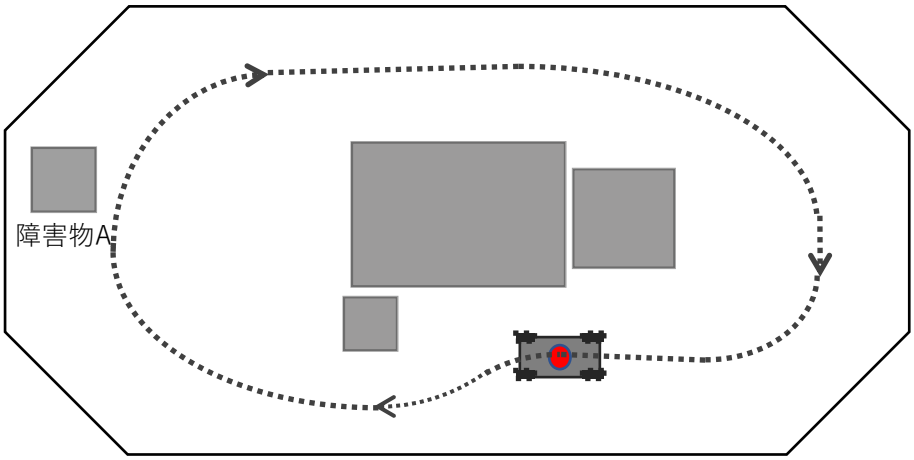
テストした状況の一例を図 4.4 に示す．図中の灰色の四角形は障害物である．このコースに対して，時計回り(図 4.4(a))と反時計回り(図 4.4(b))に走行させた．模型自動車が走行した経路は図中の点線のようになった．この実験では，模型自動車の走行経路を決定する判定円の半径を 500㎜ から 1500㎜ の範囲で調整した．その結果，次のようになった．

① 500㎜ の半径で制御すると，ステアリングの切り替えが頻繁に発生し，右往左往する動きになる．小回りが利いていると言えなくもないが，ぎこちない印象であった．

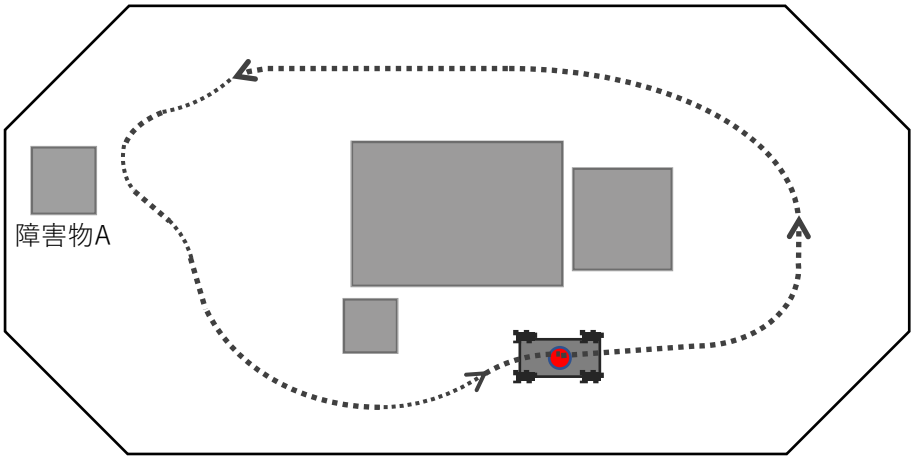
② 1000㎜ の半径で制御すると，ぎこちない動きが減り，滑らかに走行した．

③ 1500㎜ の半径で制御すると，動きはさらに滑らかになる．しかし，4.2 で述べたように 1000㎜ の制御半径で避けられる障害物が避けられなくなる．

以上のことから，今回の研究では 1000㎜ の判定円で制御する場合が最適であると判断した．



(a) 時計回りに走行させた場合



(b) 反時計回りに走行させた場合

図 4.4 周回コースの概要と時計回り・反時計回りの軌道

4.4 試作システムの性能

完成させた模型自動車の特性値を表 4.1 にまとめる．模型自動車の速度は，前進方向に 30 段階，後退方向に 30 段階で制御可能である．今回は設定可能な最低速度で前進走行だけをさせた．ステアリング角は-15～15 の整数で設定可能であり，これらの数値に対応して，概ね-15° ～15° の範囲で前輪のタイヤ角度が変化した（詳細は表 3.1 を参照）．LIDAR センサの距離の測定範囲はおおよそ 10m である．LIDAR センサは 360° の範囲で 297 点の距離を計測する．開発したプログラムでは，それを 360 点にリサンプリングして利用した．LIDAR センサ自身の計測速度は 12～13 回転/秒である．模型自動車を自律走行させた場合の制御ループの速さは 0.08 秒であった．その内訳は，LIDAR センサからのデータ入力に 0.073 秒，グリッドマップの生成に 0.003 秒，目標点を計算し，制御することに 0.004 秒であった．

表 4.1 模型自動車の特性値

速度	設定可能な最低速度(300mm/秒程度)
ステアリング角の範囲	±15° の範囲を-15～15の整数で制御
LIDARセンサの距離範囲	0.15m～12m
LIDARセンサの計測点数	297点/360°
LIDARセンサの計測速度	12回転/秒
制御ループの速さ	0.08秒/ループ

4.5 考察

4.1～4.3 節の実験から，局所的な目標点を定めそこに向かう経路を逐次決定することで，模型自動車をある程度自律走行させることができた．ただし，次のような課題が残っている．

① 利用した模型自動車で曲がりきれない経路ではストップしてしまう．バック走行とハンドルの切り返しを実装する必要がある．

② 利用した LIDAR センサの特性上，赤外光を吸収する黒色や反射率が低い素材(ガラスや光を乱反射する性質のあるもの)は検出しにくい．

③ 局所的に目標点を定めることで自律走行を行なっているため，大域的な目標に向かって走行させることができない．

本研究では前進走行の制御だけを扱った．しかしこれだけでは，袋小路に入り込んだ場合に脱出することができない．これを解決するにはバック走行やハンドルの切り返しが必要になる．今回の経験から，これらの機能を実装することがそれほど容易ではないことがわかった．

本研究で，利用した LIDAR センサの信頼性が高く，精度も良いことがわかった．それでも，検出できなない障害物が存在する．実車の場合には，これに対処するためにミリ波レーダなどを使うことで，性能を補っている．

大域的な目標点を設定するには，グルーバルな地図情報が必要である．技術的には Visual SLAM とよばれる技術で内部的に地図情報を構築し，それをグローバルな地図と比較しながら制御するような技術になる．

5 章 結論

本研究では，Raspberry Pi が制御する模型自動車に，LIDAR センサを用いて局所的に目標点を決定する機能を加えることで，自律走行を行えるシステムを開発した．開発したシステムは障害物を配置した周回コースを，障害物を避けながら周回することができた．

以下の問題が課題として残った．前進走行だけの制御を行ったので，袋小路などに侵入した場合に走行不能になる．制御ループごとに設定する局所的な目標点へ向かうため，大域的な目標点に向かうことはできない．そのためには，周囲の状況をあらかじめ地図として記憶させ，その地図情報を参照しながら自己位置を推測し，模型自動車の制御を行う必要がある．LIDAR センサの特性上の問題に対しては，カメラやミリ波レーダといった別のセンサを併用すること必要がある．

参考文献

[1] 矢野経済研究所, 「ADAS/自動運用センサ世界市場に関する調査を実施（2018年）」.
(https://www.yano.co.jp/press-release/show/press_id/1915)

[2] 政府 CIO ポータル, 「官民 ITS 構想・ロードマップ 2018」.
(https://www.kantei.go.jp/jp/singi/it2/kettei/pdf/20180615/siryou9.pdf)

[3] 大島樹端久, 「シングルカメラステレオを用いた模型自動車の自動運転に関する研究」, 京都産業大学コンピュータ理工学部特別研究報告書, 2017 年度.

[4] MyEnigma, 自律移動ロボットのためのグリッドマップ作成 MATLAB, Python サンプルプログラム, https://myenigma.hatenablog.com/entry/20140714/1405343128

[5] 独立行政法人自動車技術総合機構 審査事務規程,
http://www.naltec.go.jp/images/info/pdf/jimukitei/10_Shinsajimukitei_07_08_007.pdf

謝辞

本論文を作成にあたり, 丁寧な御指導を賜りました蚊野浩教授に感謝いたします.

付録 1 Raspberry Pi と Mac 間の WiFi 経由の SSH 通信手順

Mac 側の設定

・Mac に Ethernet ケーブルでネットワークを有線接続にする

・設定画面から共有を選び

・共有する接続経路を Thunderbolt Ethernet に

・相手のコンピュータでのポートを WiFi に

・WiFi のオプションを選択

・名前を入力=設定するネットワーク名

・パスワードを入力=設定するネットワークのパスワード

・セキュリティを WPA2 パーソナルに

・Raspberry Pi 側に挿されている microSD を mac で読み取り以下のコマンドで wpa-supplicant.conf を編集する

```
$vi /volumes/boot/wpa-supplicant.conf
```

以下が wpa-supplicant.conf の内容

```
---------------------------------------------------------------------------------------------
ctrl_interface=DIR=/var/run/wpa_supplicant.conf GROUP=netdev
update_config=1

network={
        ssid="設定したネットワーク"
        psk=”設定したネットワークのパスワード”
        key_mgmt=WPA-PSK
}
---------------------------------------------------------------------------------------------
```

・以下のコマンドで空の ssh ファイルの作成

```
$touch /volumes/boot/ssh
```

Raspberry pi 側の確認

・まず WiFi がつながっていることを確認

・デスクトップの右上にある WiFi の扇形のマークにカーソルを合わせると現在接続されている IP アドレスが見ることができる．または以下のコマンドの実行結果の wlan0 の項目にも書かれている

$iwconfig wlan0

・SSH の設定を有効にしておく

$sudo raspberry.conf

というコマンドから Raspberry Pi の設定画面のインターフェースより SSH を有効にする

Mac から以下のコマンドで SSH 接続

$ssh pi@(Raspberry Pi の IP アドレス)

または

$ssh pi@raspberrypi.local

付録 2 開発したプログラムの説明

[プログラム名]

main.cpp

[内容]

模型自動車に搭載した LIDAR センサから，計測データを受けとり，360 個のデータに変換する．変換したデータは直交座標に変換し，グリッドマップを構成する配列と対応づける．その後，障害物がない領域から目標点となる座標を一つに決定する．その目標点に対してそこに向かうためのステアリング角を決定し模型自動車を制御するプログラム．

[プログラム名]

control_v3.c

[内容]

模型自動車のステアリングや前進後退を制御するコマンドを送信するプログラム．

[プログラム名]

rplidar_connect.cpp

[内容]

本研究で用いる LIDAR センサへの接続を行うためのプログラム．

[プログラム名]

steering_and_radius.cpp

[内容]

模型自動車が曲がるステアリング角それぞれとその回転できる円の半径との関係を保持したデータ．