申請先学部　　工学部　採択番号　No.　　4

令和３年度大阪大学未来基金「学部学生による自主研究奨励事業」研究成果報告書

ふりがな 氏名	おおた あつし 大田 純志	学部 学科	工学部 応用理工学科	学年	２年
ふりがな 共同 研究者氏名	ながた げんき 永田 源貴	学部 学科	工学部 応用理工学科	学年	２年
	にしむら たつき 西村 樹希		基礎工学部 システム科学科		２年
					年

アドバイザー教員 氏名	ワン ウェイウェイ 万 偉偉	所属	基礎工学研究科
研究課題名	一般への普及を目的とした自動走行システムの開発		
研究成果の概要	研究目的，研究計画，研究方法，研究経過，研究成果等について記述すること．必要に応じて用紙を追加してもよい．（先行する研究を引用する場合は，「阪大生のためのアカデミックライティング入門」に従い，盗作剽窃にならないように引用部分を明示し文末に参考文献リストをつけること．）		

1　研究目的

昨今のコロナ禍において，人との対面を避ける機会が多くなったことからソーシャルディスタンスが求められ，非対面でのコミュニケーションの需要が高まっている．また，最近ではロボットが自動で接客を行う飲食店ができるなど，単純労働の削減を目指す新しい社会において，自動で動作を行うロボットの需要はますます高まっている．しかし，自動動作，特に自動走行の開発には，開発環境として，ソフトウェア開発を行う企業を除く一般の企業や学校で主に使用される Windows，MacOS 等ではなく，Ubuntu などの開発者に向けた OS が主流として使用されている．また，自己位置推定，経路計画，経路追従など様々な技術が必要であることから，一般への自動走行ロボットの普及は容易ではない．そこで，専門的な知識，技術や動作環境がなくともロボットを操作できる GUI を作成すること，また手動操作と同等以上の精度を持つ自動走行を実現することを研究目的とした．

2　先行研究[1]の要約

先行研究として，過去に自主研究で行われていた「自走式ロボットによる環境マッピングと自己位置推定」が挙げられる．今回の研究では“より広く一般に自動走行ロボットが使用される”ことを目標としているため，先行研究は足回りが対向二輪であるのに対し，限られたスペースでも方向転換ができるという点から三輪オムニの足回りを用い研究を行った．また，先行研究では SLAM（Simultaneous Localization And Mapping）を行う手段として，LRF と呼ばれる測距センサとオドメトリ*1 による gmapping が用いられているが，今回の研究ではより高度かつ容易に SLAM を LRF のみによって行う Google Cartographer を用い研究を行った．

3　研究準備

今回の研究では，手動操作を行うための GUI コントローラーを作成し，GUI コントローラーを用いた操作と GUI コントローラーを用いない操作の評価を行った．次に，自動走行の実験を行うために TurtleBot3[*2] を用い自動走行を行うために必要な知識を身に付け，後に製作した機体を用いて実験を行った．本章では実験を行うにあたり準備した事柄に関して説明を行う．

3.1　TurtleBot3 を用いた ROS[*3] の学習

一から自動走行を行うことは非常に困難であるため，自動走行を行うために必要なパッケージが存在する ROS を用いて自動走行を行うことを計画した．製作した三輪オムニ機体での ROS を用いた自動走行を行うための前哨として，ROS を学習するための教材である TurtleBot3 を用いて自動走行を行うために Localization[*4]，Mapping[*5] これらを複合した SLAM，Navigation[*6] についての学習を行った．

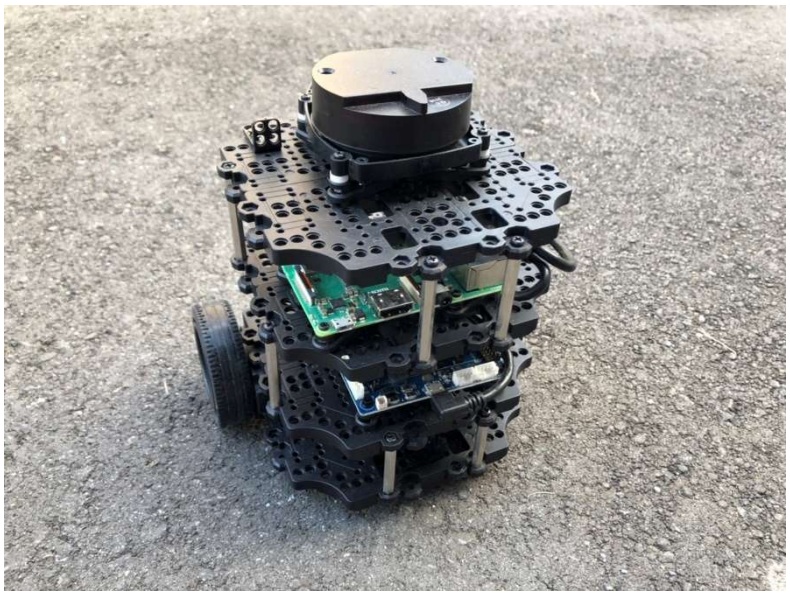


図１　今回使用した TurtleBot3

3.2　機体設計・製作

今回の研究では，人間が通行する道や機体が通過するために必要な幅がわずかしかないような道を走行することを想定し，方向転換をその場で行える三輪オムニを用いた．下図 2 のような機体を設計，製作した．三輪のオムニで地面に対して設置しているため，いずれかのオムニが浮くことなく安定した走行を行うことが可能である．

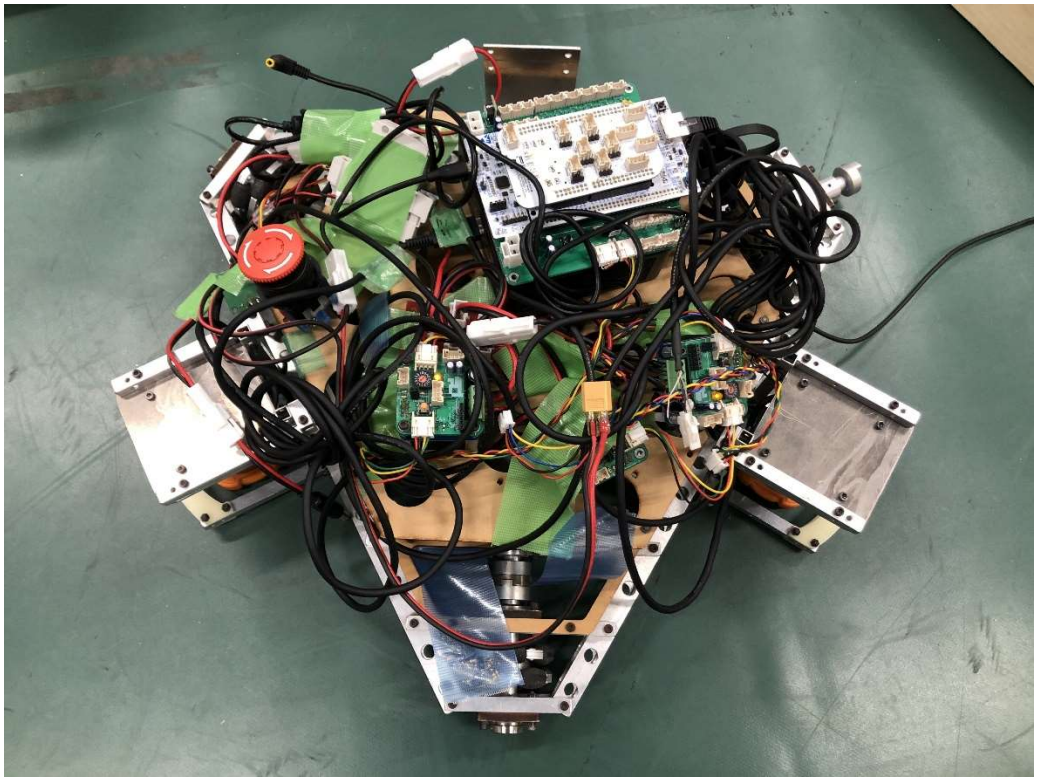


図 2 今回製作した三輪オムニ型の機体

3.3　回路基板・通信

今回の研究においてメインとなる PC は ROS の実行が可能かつ機体に搭載可能な INTEL NUC を，回路基板は評価ボードとして NUCLEO-F767ZI（以降，F7 マイコンと呼称する）を，モーターを制御するために CAN 通信*7 を行うことができる自作の基板を用意した．

以上の機器を用いた通信関係図は下図 3 で表される通りである．

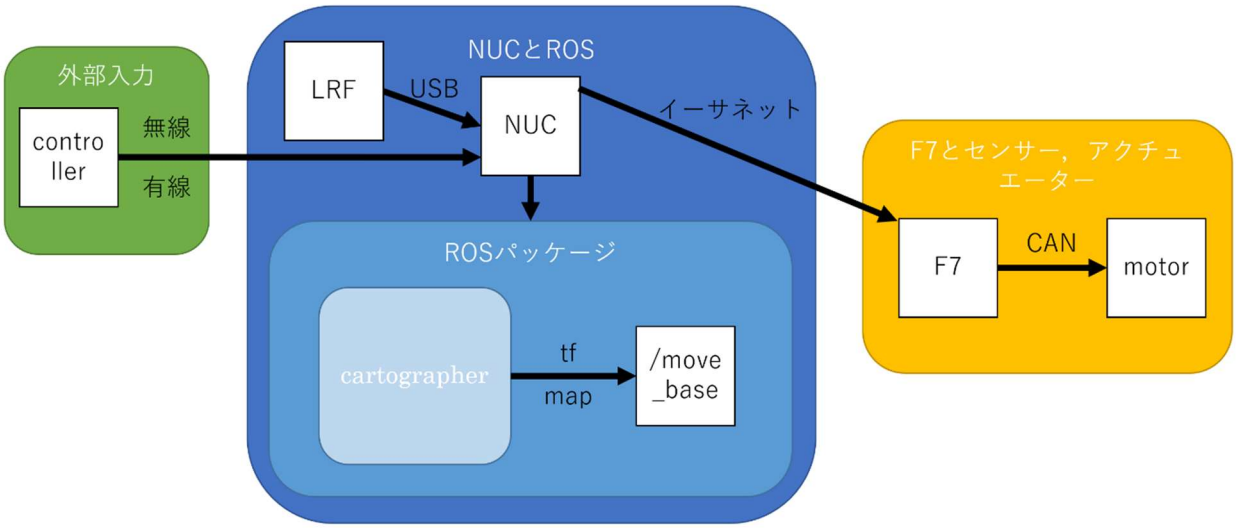


図 3 作成した通信関係を表す関係図

3.4　手動操作のための GUI コントローラー作成[3]

はじめに自動走行を行う GUI を作成することが非常に困難であったため, 今回の研究では Web 上でロボットを操作可能なコントローラー機能を有した GUI の作成を目標とした．このコントローラーは操作した際に発生する値を ROS 上のデータとして送り機体に速度指令を送信する役割を持つ．一般的なコントローラーに用いられている Joystick と呼ばれるコントローラーを採用し，作成したサイトの URL にアクセスすることで誰もが直観的かつ容易に，複数のコントローラーを必要とせずに同時にロボットを操縦することが可能な GUI コントローラーを作成した．この web サイトにおいてフロントエンドの作成には Angular*8 を用い，バックエンドには ros-bridge-server*9 を用いた．

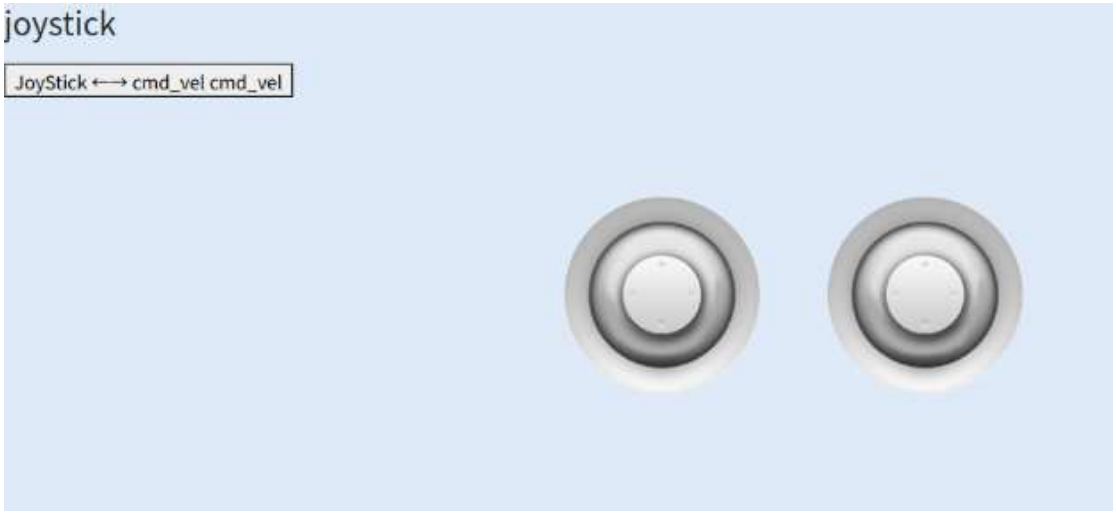


図 4 作成した手動操作のための GUI コントローラー

3.5　自動走行のための環境構築[4][5]

今回の研究では上記 rqt_graph と呼ばれる ROS 上のデータフロー図を表示する GUI プラグインにおいて示される ROS の環境を構築した. その中でも特に, 今回の自動走行の研究に関連するシステムを下記で説明を行う.

まず，Google Cartographer によって SLAM を行った．今回の研究では北陽の LRF を用いた Localization，Mapping を行うために，機体のパラメータを測定し調整を行った．

次に，Google Cartographer からの情報を処理し実際に経路生成及び経路追従を行う node*3 である/move_base（下図 5 中央部）を用いた．move_base とは，自己位置推定の情報と地図の情報を受け取り，現在地から指定した位置までの経路を生成し追従するための速度指令を送るシステムである．

最後に，メインの PC と F7 マイコンとの通信を行うシステムを作成した．図 5 右下にある/cmd_vel_to_F7_node から/udp_send_node 部分が通信を行う箇所である．これらの node は ROS と F7 マイコン間の通信を，UDP 通信*10 を用いて接続し，/move_base から出力された機体の速度情報である/cmd_vel topic*3 を送る機能を持つ．これらの node を作成することで ROS 上でのデータを F7 マイコンに送り，機体を動かすことが可能となる．

以上についてまとめると下図 5 のように示される．

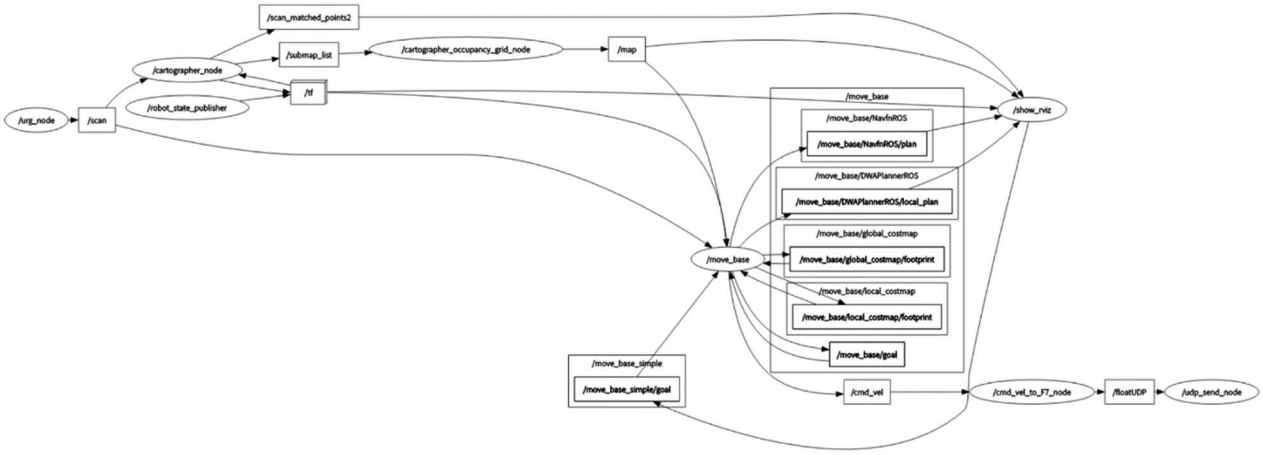


図 5 ROS 上で構築したデータフローの図

4　作成した GUI コントローラーの評価

今回の研究で作成した GUI コントローラーと市販のゲーム用コントローラーを用いて機体操作を行ったケースの２パターンの評価を以下の点に着目し比較した．

4.1　利便性

Web 上でロボットを操作可能なコントローラー機能を有した GUI を公開した後に，スマートフォンやタブレット，PC などの JavaScript*11 を実行可能な Web ブラウザが動作する電子機器でアクセスすることで誰でも機体を操縦することが可能である．一方，ゲーム用コントローラーではメインの PC から操作が可能である．前者は同一の LAN 環境下であれば場所の制限を受けないのに対し，後者は通信距離の制限が存在する．また前者は一般的な電子機器があればよいのに対し，後者は機体を操作するための外部入力を行うコントローラーが必要になる．このことから，利便性の観点からは今回の研究で作成した GUI コントローラーの方が優れていると考えられる．

4.2　使用難度

今回の研究で作成した GUI コントローラーとゲーム用コントローラーはどちらも Joystick を採用している．このことから，コントローラーに関して使用難度の差が発生することはないと考えられる．

しかし，前者は自身の電子機器から OS を問わず，公開された Web サイトから操作可能であるのに対し，後者は Ubuntu などの ROS を実行可能な OS からコマンドライン上で操作し起動しなければならない．今回の研究の目的である誰でも使用可能なシステムの開発を実現するためには，専門的な知識を必要としない GUI コントローラーの方が優れていると考えられる．

図 6 コマンドライン上で ROS を操作する様子

4.3　操縦性

今回の研究で作成した GUI コントローラーを用い機体を操作した際に, 認識可能な程度の通信でのタイムラグが発生していることが確認され，このことが原因で機体の操作に若干難があることが分かった．対してゲーム用コントローラーを用いた場合は通信でのタイムラグは人間では認識不可能であった．このことから，操縦性の観点においてはゲーム用コントローラーの方が優れていること考えられる．

図 7 操作の様子と Joystick の反応の様子

5　実験

以上の内容を用い，作成した自動走行機体の Mapping, Localization, Navigation の評価を行った．

5.1　実験機体

3.2，3.3，3.5 に記載した自動走行を行うことが可能な環境を構築した機体を用いて本実験を行った．

5.2　理論

構築した ROS 環境では move_base node から送られる cmd_vel topic を，今回作成した /cmd_vel_to_F7_node，/udp_send_node を通じて F7 マイコンへ機体の速度情報を送る．しかしながら，cmd_vel topic の速度の記述形式が(x, y, ω)*12 であるのに対し，F7 マイコンがモーターへ指令を行う際の出力の記述形式は各一つあたりのモーターに対して電圧の値を送るというものであった．

i. x, y, ω方向のみに回転する仮想モーターをそれぞれ設定する．

ii. コントローラー入力の際の一方向の最大出力を m/s の単位で測定する．

iii. ii の速度に対する/cmd_vel node の速度の割合を算出する．なお，これが i で仮定したx, y, ω方向のみに回転する仮想モーターの出力値となる．

iv. 実際に用いる機体は 3 輪オムニであるため，x, y, ω方向に動くときの必要な実際の各モーターの出力値をそれぞれ算出する．

v. x, y, ω方向それぞれの iv の計算結果の和をモーターごとに算出する．

vi. vi で得られた計算の値を機体の自由度 3 で割る．これは vi の計算結果の最大値が機体の自由度と同じく 3 となり，各モーターの一方向の出力の最大値を 0.333...とするためである．

vii. /cmd_vel node で送られる速度は ii で計測した速度よりも小さいと仮定する．これは事前に topic を確認した際に， ii の値を超える指令は送られてこなかったためである．

ここで，実際に測定を行う様子を下図 8 に示す．

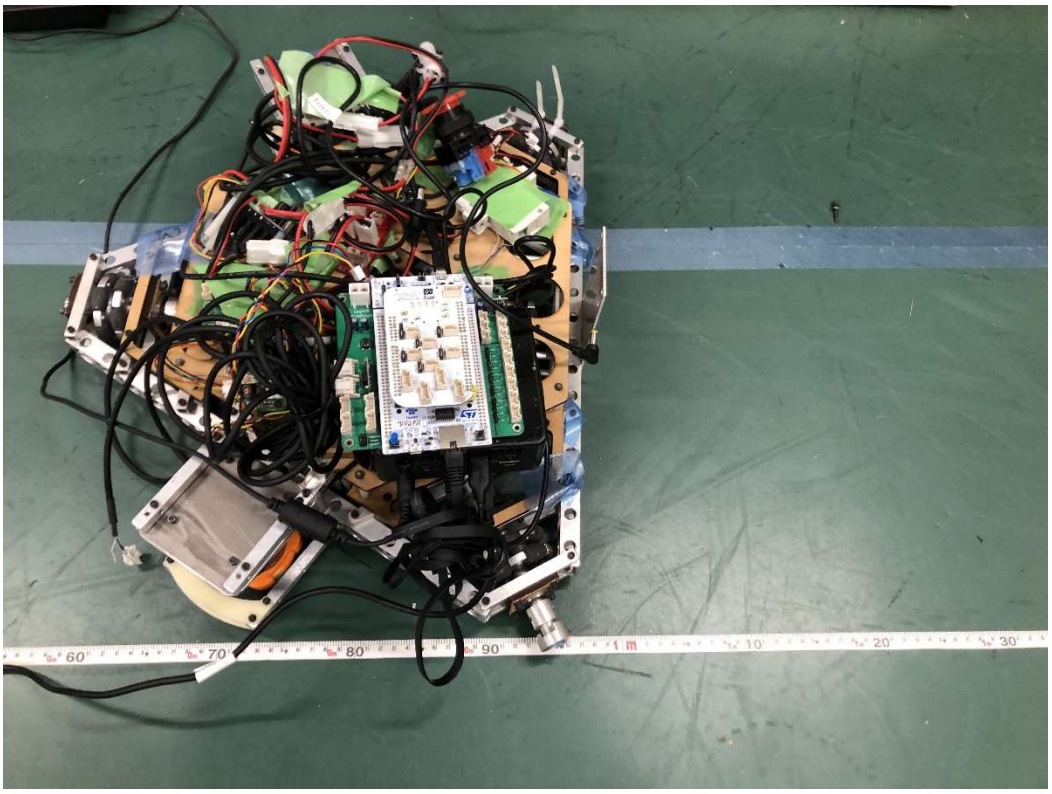


図 8 速度の計測を行う様子

5.3　手順

rviz で Navigation の情報を可視化して本実験を行った．まず，下図のような機体の初期位置（画像右下部）と目標位置（画像左下部）が一本の直線では結べないような障害物の配置されたフィールドを作成した．このフィールド上で初期地点から目標位置まで移動することが可能であれば直進，回転方向に対して正常な動作を行っていることが確認されるため，今回はこのようなフィールドを作成し実験を行った．

図 9 作成したフィールド

5.4　結果

作成したフィールドに対して Mapping されたフィールドは下図 10 のようになった．画像を確認する限りでは大きな測定誤差は見られなかったため Mapping に関しては正常に動作を行っていることが確認された．

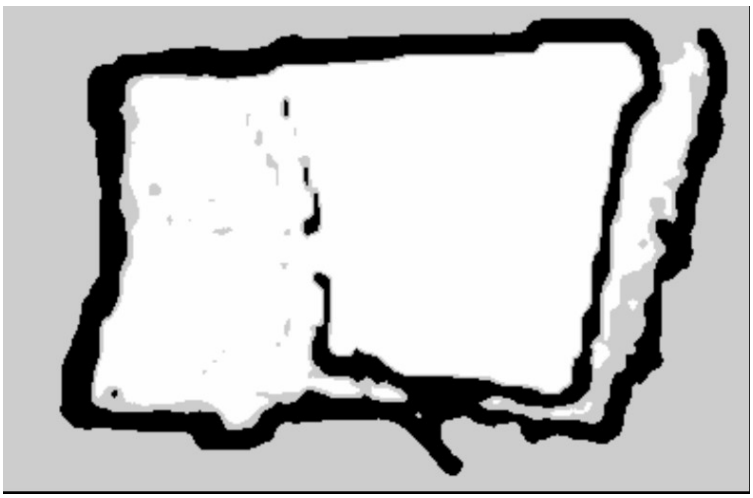


図 10 生成された地図

次に，Localization について，こちらもψ方向に回転を連続で数十秒行う，人や壁と激突する，機体に対して横転する様な過度な衝撃を加えるなど機体に対し高い負荷を加えない限りは正常な動作を行っていることが実際の走行及び rviz 上から確認された．

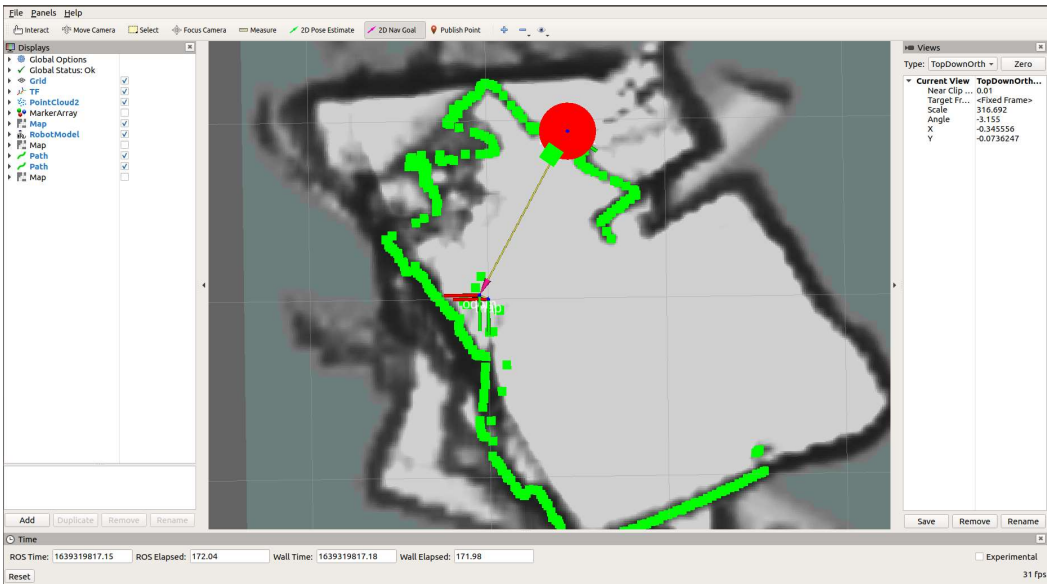
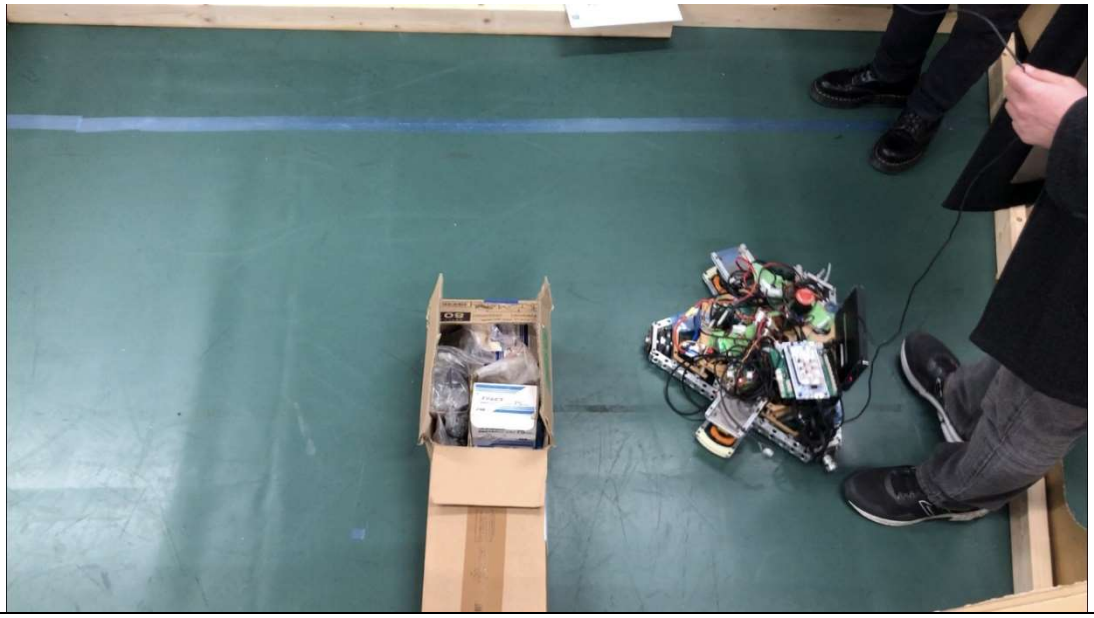
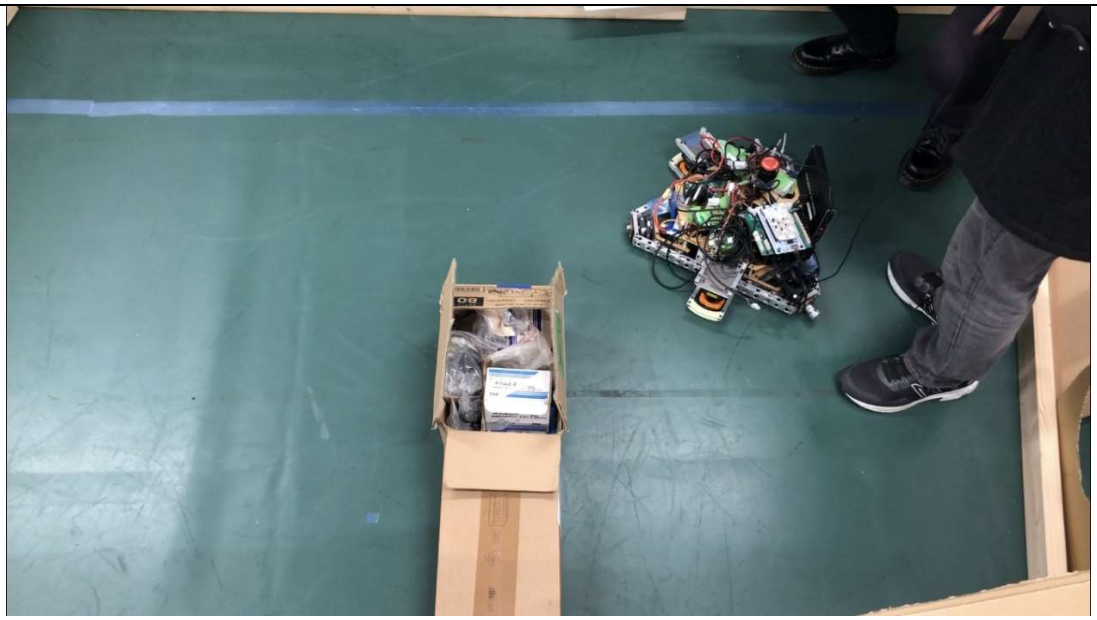
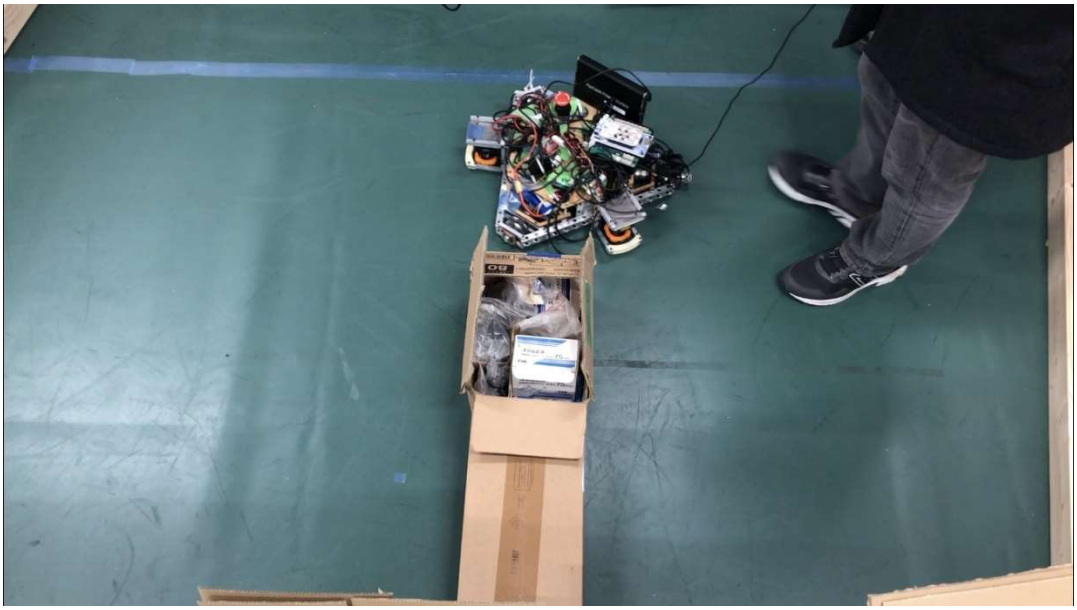


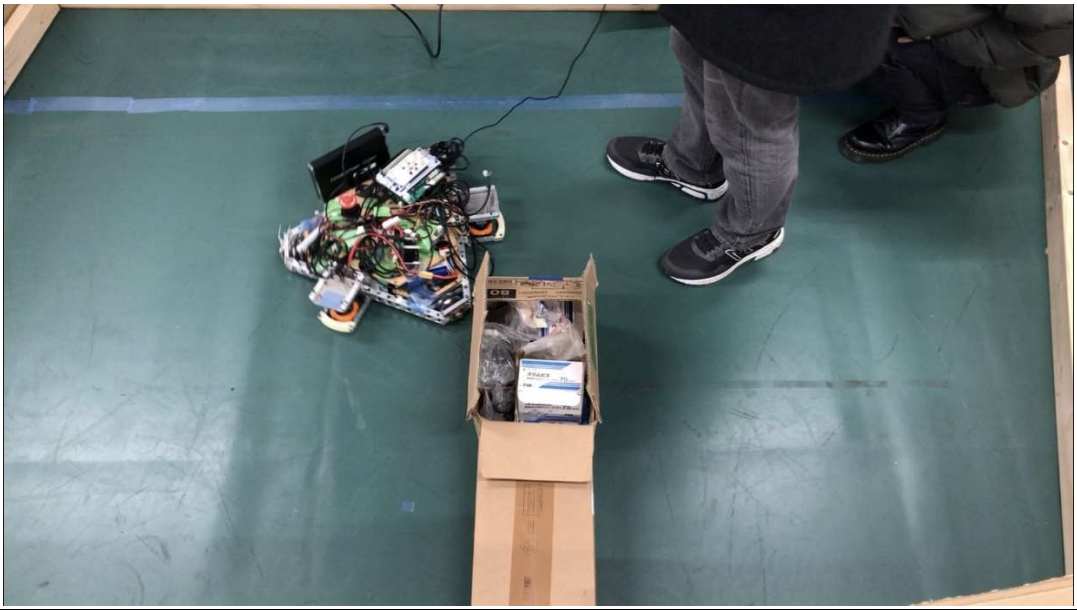
図 11　自己位置推定を行う様子

最後に，実際の動作に関して，下図 12 から下図 16 が実際に自動走行を行っている様子である．初期位置から目標位置まで自動走行を行った結果，初期位置から目標位置まで正常に動作を行うことが確認された．しかしながら，今回我々が使用した Navigation の機能上の問題として，地図上の詳細な地点を設定した自動走行を行えなかったことから，rviz 上での走行と実際の走行の誤差を比較することができなかった．









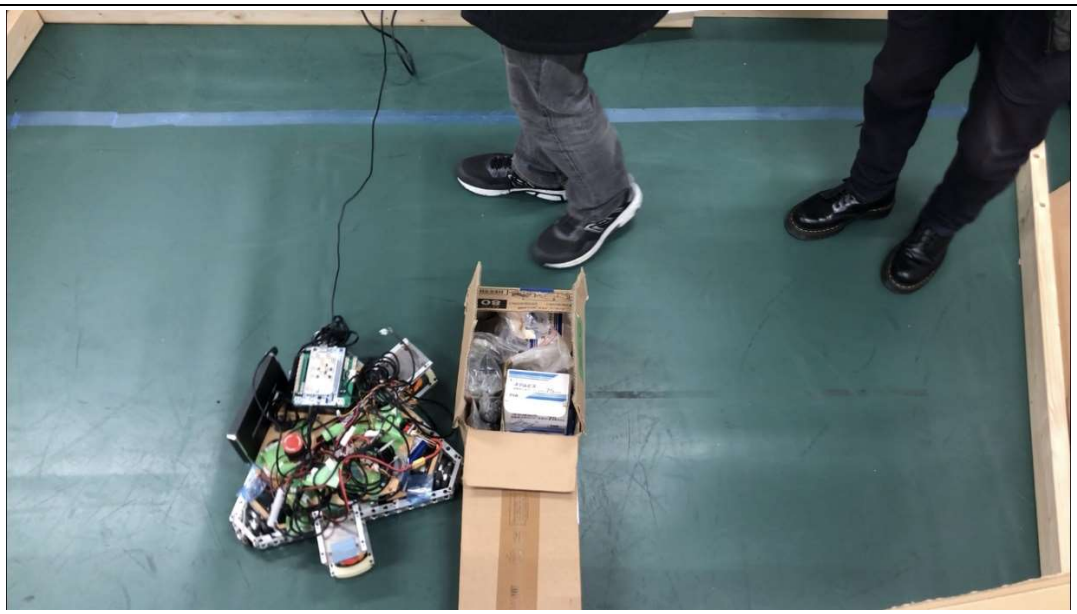


図 12,13,14,15,16 ロボットが初期位置から目標位置まで自動走行を行う様子

5.5　考察

5.2 の理論より，モーターの電圧を求める計算において，一つ目は電圧が常に一定であるという近似，二つ目は今回使用したモーターの性質を，出力可能な電圧の最大値に対して，十分長い距離を走行した際に速度が一定値に達した条件下であればある一定と近似した電圧の値の割合を機体の速度と近似すると仮定した．一つ目の近似に関しては実際に電圧を測定したとしても測定誤差が発生すること，また速度に大きな誤差を及ぼさないことが推定されるため，近似をしても問題ないと考えられる．加えて，二つ目の近似に関しても“十分長い距離において速度が一定値に達した条件下であれば”という条件を設定しているため，モーターの性質を加味し近似が可能であると判断した．

しかしながら，計測の誤差などを鑑みるに Navigation を行った際に少なからず誤差を発生していることが予想される．具体的な数値を出すことは困難であるが，今回の実験機体の最大速度が 0.1814m/s であり，速度を測定する際に計測した時間の平均値との差の最大値が 0.36 秒であったことから，モーターにかかる電圧を計測した段階で 5cm~10cm 程度の誤差が発生していることが推測された．

6　今後の展望

5.4 の実験の結果にもある通り，Navigation を行った際の誤差の値を算出できなかったことが実験の課題として挙げられる．5.5 の考察にもある通り，最低でも 5~10cm の誤差が発生している可能性が考えられるため，まずは誤差の値を算出することがより精度の高い自動走行の評価に繋がると考えた．

また，今回の研究では手動走行の GUI コントローラーの作成，また自動走行を行う GUI の導入を行った．しかしながら，後者は実用に至るような性能ではなかったため，次回はより高度な自動走行の GUI の作成に挑戦したい．

以上二点を今後の研究課題とし，この度の研究を締め括る．

7　謝辞

本研究を進めるにあたり，ご指導を頂いたアドバイザー教員の万偉偉先生に感謝致します．また，実験機体製作，並びに実験フィールドの展開にご協力して下さった大阪大学 Robohan の皆様に感謝します.

8　参考文献

[1] 勝田 充輝，丹羽 英人 自走式ロボットによる環境マッピングと自己位置推定
2018-04 http://hdl.handle.net/11094/68121
[2] ROBOTIS e-Manual https://emanual.robotis.com/
[3] Qiita roslibjs の基本的な使い方
https://qiita.com/daigakupotato/items/bb44c070d4304caa4bf1
[4] 表允哲 倉爪亮 ジョンジョウン 2018 ROS ロボットプログラミングバイブル オーム社
[5] Qiita Cartographer で Navigation を行おう!
https://qiita.com/MMM-lab/items/14e94f9d41ccca4e1c61
また，ROS の基本操作，環境構築においてトランジスタ技術 9 月号を参考文献とした.

9　注釈

*1 車輪の回転角を計算することでロボットの現在位置や姿勢を推定する方法のこと.
*2 ROBOTIS 社が販売する移動ロボットプラットフォームのこと.
https://e-shop.robotis.co.jp/list.php?c_id=93
*3 Robot Operating System の略．ロボット用のソフトウェアプラットフォームであり，Ubuntu などの Linux を中心に動作する．ROS 上では 1 つのプログラムは node と呼ばれ，node 同士は pub/sub 通信で topic と呼ばれるデータをやりとりする.これらを組み合わせることにより，ロボット内のシステムを構成することができる.
*4 ロボットが 1 つないしは複数のセンサを用いて自分自身の位置を推定すること. 日本語では「自己位置推定」とも呼ばれる.
*5 LRF などセンサを用い地図を作成すること.
*6 自己位置推定，経路計画，経路追従などを組み合わせて現在位置から指定された位置まで自律的に移動すること.
*7 Controller Area Network の略. Bosch 社が 1985 年に車載ネットワーク用に開発した通信規格であり，最大 1Mbps 程度の速度でマイコン間の通信を行うことができる.
*8 フロントエンド Web アプリケーションフレームワークの一種．Google 社などにより開発された.
*9 ROS と Web サイトで通信を行うための ROS のパッケージのこと.
https://github.com/RobotWebTools/rosbridge_suite
*10 UDP（User Datagram Protocol）と呼ばれるトランスポート層で動作するプロトコルを用いた通信方法のこと.
*11 プログラミング言語の一種．Web ページで多く用いられる.
*12 ロボットのx方向とy方向の並進速度であるx（m/s），y（m/s）とψ角方向の回転速度であるω（rad/s）の 3 つの値を組み合わせたもの.