

# GNSS と 2D 地図の位置推定切替機構を備えたクローラロボットナビゲーションシステム

中村 勇太<sup>1</sup>, 吉田 侑樹<sup>1</sup>

<sup>1</sup> 株式会社 CuboRex

## 1 はじめに

近年、労働人口の減少や働き方改革により現場での作業の効率化が求められている。こうした背景でロボットには工場や倉庫での運搬作業やビルなどの警備、屋外では工事現場の清掃や農薬の散布などの作業が自動化されることが期待されている。特に、Geek+社のEVEシリーズ[1]やLexxPluss社のHybrid-AMR[2]のようなAGVやAMRは大量生産を行う工場や大規模な物流倉庫での運搬の自動化を進め、SEQSENSE社のSQ-2[3]やugo社のugoシリーズ[4]のような警備ロボットでは実際のビルの警備を少ない人数で運用することに成功している。これらのロボットでは、2D/3Dの違いはあるが使用環境で地図を作ることで自律走行を実現し、場所は限定されているが今日でも現場の効率化に絶大な効果を与えていている。

しかし、発電所の除草作業や工事現場の運搬作業など、屋外の現場においては大半が自動化されておらず、ラジコン操作や遠隔映像による操作のロボットで実証実験を行うところまでしか進んでいないという現状がある。以上から、屋外でのロボットによる仕事の効率化や協業、代替に対してはまだまだ発展途上であると言える。

CuboRexでは、一輪車を電動化した“E-cat kit2”[5]や走破性の高いクローラユニット“CuGo”シリーズ[6]をリリースし、“現場のツライをロボティクスで改善する”を掲げ、屋外での現場効率化を進める活動をしている。この“CuGo”シリーズにロボット制御ミドルウェアである“ROS”でのアプリケーション開発ができる環境を整えることで、AMRと呼ばれるロボットの活動範囲を広げることができ、屋外の現場における自動化がより一層発展できると考えた。殊に、つくばチャレンジでは、リアルワールドでの自律走行技術レベルを向上することを

目標としていることから、つくばチャレンジでの課題を“CuGo”で実現することが“現場のツライをロボティクスで改善する”の近道となると考え、つくばチャレンジに参加した。

本年度のつくばチャレンジの取り組みでは、“ROS開発キット CuGo V3”[7]をベースに走行用ロボットを作成し、GNSSと2D地図の位置推定を切り替えることができるナビゲーションシステムを開発した。GNSSでは、CLAS[8]を利用することにより基準基地局なしのロボットの単独測位で非常に高精度な位置を取得できたため、この値をそのままロボットの自己位置とした。本年度から屋根があるエリアが追加され、GNSSの測位結果のみでの走行が難しい区間が設定された。この区間は従来どおり2D地図を作成し位置推定することで走行し、両者の自己位置の入力を切り替えられるようにした。これにより、高精度な衛星測位が可能な屋外環境だけでなく、衛星測位を遮る屋根のある区間も安定して走行することができるロボットとなった。

本走行では、確認走行区間を抜けた先のパイロン地帯でパイロンの前に停止して断念した。計算資源が足りなかったことと、設定した経路が未熟であったため、もう少し距離を伸ばすことができると考えられる。本年度では、“ROS開発キット CuGo V3”を使用し、2DLiDAR、GNSS、ホイールオドメトリのみを使ったシンプルなナビゲーションシステムで幅広い環境で走行できるシステムを構築することができた。

## 2 ハードウェア

本年度の実験に使用したロボットの外観を以下の図1に示す。ハードウェアの構成を表1に示す。つくばチャレンジ走行用ロボットには、CuboRex製自律走行ロボットの開発用プラットフォームである“ROS開発キット CuGo V3”をベースに、課題達成

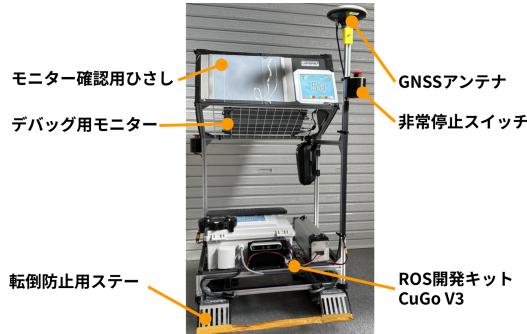


図 1 つくばチャレンジ仕様 CuGo



図 2 ROS 開発キット CuGo V3

表 1 ロボット諸元

前長	0.7 [m]
全幅	0.7 [m]
全高	1.3 [m]
重量	33 [kg]
最高速度	3.5 [m/s]
駆動機構	クローラ
制御 PC	Jetson Orin NX 16GB
モータ制御用コントローラ	Arduino Uno R3

に必要な装備を追加した。

## 2.1 ROS 開発キット CuGo V3

“ROS 開発キット CuGo V3”は、汎用クローラユニットである CuGo V3、それを正確に制御するマイクロコントローラとして Arduino UNO R3、ROS を実行する Linux コンピュータとして Jetson Orin NX、測域センサとして RPLiDAR S2E、そして、それらを駆動するバッテリーや電源システムがオールインワンとなったパッケージ製品である。これに必要に応じて 3D LiDAR や GNSS、カメラなどの装備を加えることで、最小限の労力でクローラロボットの開発を開始することができる。

## 2.2 つくばチャレンジ仕様

本年度のつくばチャレンジでは、“ROS 開発キット CuGo V3”に GNSS、デバッグ用モニター、モニター確認用のひさし、転倒防止用のステーを追加し、機体拡張のため押しやすい位置に非常停止スイッチを再設置する変更を行った。後述するが、ナビゲーションシステムに使用したセンサは 2DLiDAR、GNSS、ホイールオドメトリのみという非常にシンプルな構成となっている。計算資源も前述の “ROS 開発キット CuGo V3”に搭載されたもののみで実現可能なため、比較的安価で製造しやすい屋外走行ロボットとなった。

## 3 ナビゲーションシステム

### 3.1 ナビゲーション概要

本年度のつくばチャレンジのナビゲーションシステム構成図を図 3 に示す。このシステムでは、基本的にみちびきのセンチメーター級測位補強信号を使って補正した GNSS の測位結果を使用して目的地を目指す。2023 年のコースから、GNSS の信号を遮る屋根や建物の直下を通る、市庁舎の北側から東側にかけての区間がコースに追加された。このエリアを通過するとき、図 4 のように GNSS の測位精

度の悪化を確認できたため、このエリアに限って地図を作成し地図に対して位置推定を行った。(以下、図 5 の通り、この区間を”GNSS 低精度エリア”とする。) GNSS 低精度エリアを通過した後、ふたたび GNSS の位置推定に切り替えて走行することにより、大規模な地図を作成することなく長距離の自律走行を実現した。この章では、以下の順番でナビゲーションシステムの詳細について説明する。

- GNSS による位置推定
- 2D 地図による位置推定
- GNSS と地図の位置推定の切り替え
- センサフュージョン
- Waypoint の設定
- 経路計画
- 障害物回避
- その他開発支援ツール

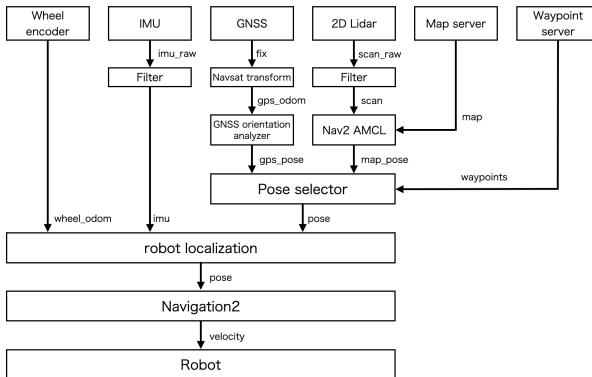


図 3 ナビゲーションシステム概要図

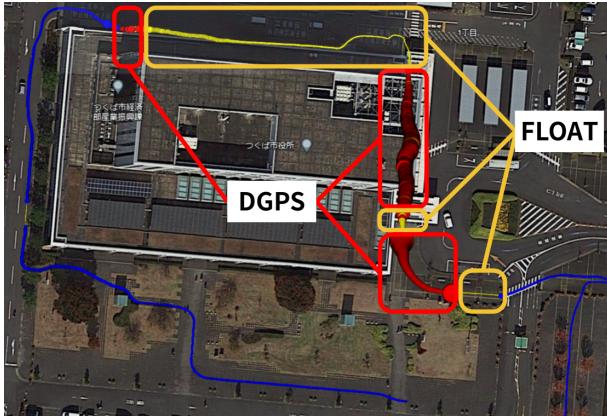


図 4 確認走行区間の測位精度悪化の状況

### 3.2 GNSS による位置推定

ロボットの内部に CLAS 受信モジュール D9CX1 と RTK モジュール F9PX1 を、機体上部に GNSS アンテナ JCA228E をそれぞれ 1 組搭載し、CLAS に

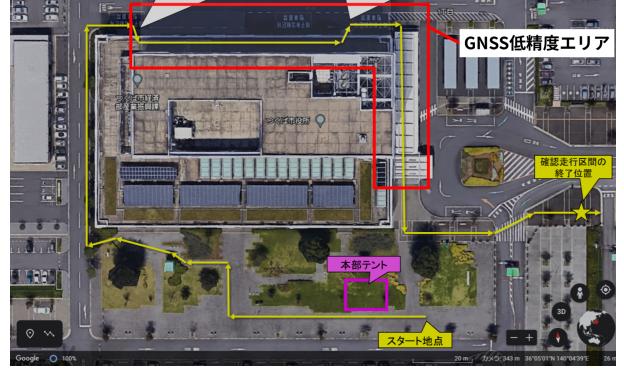


図 5 GNSS 低精度エリア

よる測位を 5Hz で行った。CLAS とはセンチメートル級測位サービスを示し、準天頂衛星みちびきが配信する補強信号を受信することで、日本国内では精度の高い測位が可能となるサービスである。CLAS の移動体における水平方向の測位精度は、12[cm](95%) 以下である。つくばチャレンジにおいては、本走行コースのゴール地点の緯度・経度(36.0826686, 140.0775115)を原点として、測位された緯度・経度との差分を自己位置とした。ロボットの姿勢角推定は、図 6 のように前回時間からの移動量をもとに算出した姿勢を自己位置とした。ただし、精度向上のため姿勢角の算出は下記の条件を満たす際のみ適用し、それ以外の状況では姿勢角を算出しなかった。

- ロボットの直進速度が 0.3[m/s] 以上
- ロボットの旋回速度が 0.3[rad/s] 以下

姿勢角が算出されない期間における姿勢角の位置推定は、後述のセンサフュージョンにより補間した。

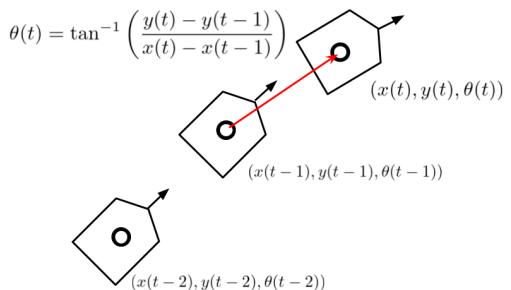


図 6 GNSS による姿勢角推定

### 3.3 2D 地図による位置推定

ロボット下部に取り付けられた 2DLiDAR を使って図 5 のつくば市庁舎周辺の地図を作成した。2DLiDAR は地面から 120mm の位置に取り付けてあ

り、周囲 360 度、最大 30m までの距離を得ることができる。SLAM は Cartographer[9] を利用した。確認走行区間のルートを 1.5 周したセンサデータを使い、パラメータを調整して地図を作成した。作成した地図は図 7 に示す。実際のナビゲーションでは、GNSS 低精度エリアとその前後 5m 程において、この地図から求められる自己位置を使用した。走行する際の地図の位置推定には、nav2\_AMCL を利用した。(場合によってカットする) Cartographer で地図を作成する際にループクローズが発生するが、そのパラメータは GNSS 低精度エリアの直線部の精度が最も高くなるように調整した。この調整をしないと、ループクローズしたときの誤差の勾配がちょうど北エリアで詳細するため、GNSS による自己位置と地図による自己位置にずれが発生する。



図 7 作成した確認走行区間の 2D 地図

### 3.4 GNSS と 2D 地図の位置推定の切り替え

GNSS と 2D 地図の位置との切り替えのため、Pose Selector を作成した。ナビゲーション構成図の Pose Selector が示す部分で上記 3.2 と 3.3 の各位置推定をした値を排他的に選択し、EKF に送る。こうすることで、センサフュージョンをする側のノードでは、現在の状態が GNSS であるか 2D 地図の位置であるかを意識することなく機械的に処理することができる。それぞれの状態は次に向かう Waypoint のデータ構造にフラグとして内包しており、Waypoint ごとにどちらの位置を使用するかを記録することで切り替えを実現した。

### 3.5 センサフュージョン

このナビゲーションシステムでは、図 3 のようにロボットの位置推定にホイールオドメトリ、IMU、

GNSS、2D 地図の位置を入力したセンサフュージョンにより位置推定を行った。IMU は 9 軸で地磁気を利用して絶対角を取得していたが、市庁舎北エリアの急速充電器付近で非常に強いノイズを受け、GNSS 低精度エリアでの姿勢角の精度悪化が無視できなかった。そのため、IMU のセンサデータは共分散を非常に大きな値に設定していて、実質使用していない。

ホイールオドメトリの値は、作動二輪モデルから \nav\\_msgs \Odometry の値を算出して利用した。GNSS の姿勢は前項の通り、直前の連続した位置からロボットの向いている方向を推定し \nav\\_msgs \Odometry に格納した。AMCL の位置推定結果は \tf での出力は無効化して、\nav\\_msgs \Odometry に変換して利用した。

3 つの \nav\\_msgs \Odometry を robot\_localization[10] パッケージを利用して EKF 処理をし、ロボットの自己位置としてナビゲーションに利用した。

### 3.6 Waypoint の設定

Waypoint の設定は、3.5 で推定したロボットの自己位置をそのまま記憶するツールを作成し、使用した。GNSS と 2D 地図の位置推定のどちらの状態であっても、先述の通り、ロボットが認識する自己位置は変わらないため、現在認識している自己位置をそのままキーボード操作をトリガーに csv ファイルに記憶する。実験走行では、当日の午前にロボットを走行させたい経路の通りにコントローラでマニュアル操作し、2m おきの Waypoint を記憶させた。午後に検定を実施し、作成した Waypoint 列の csv ファイルをナビゲーションシステムに読み込ませて自律走行を行った。

### 3.7 経路計画

ロボットの自己位置から Waypoint までの走行経路の生成は、ROS 2 標準の Navigation2[11] をそのまま利用した。Waypoint までの経路は A\*アルゴリズム [12] で算出し、その経路を RegulatedPurePursuit[13] で追従した。

### 3.8 障害物回避

障害物の回避には 2DLiDAR を使用し、ROS 2 標準の Navigation2[11] によるものをそのまま利用した。しかし、2DLiDAR を使用しているためパイロンのような起伏のある形状の障害物を回避できない

ことがある、経路生成が行われるより前に衝突する箇所に障害物が発生した際に避けられない、という問題があった。そのため、前述の機能に加えて、経路生成より高速で処理が可能な障害物判定機能を追加した。具体的な処理は、コストマップで機体前方の点のコストを常に参照し、定点のコストが閾値以上となった際には停止させるものとした。この閾値を低くすることで障害物に対する接近を抑制することが可能となり、障害物が接近した時点で停止することで、起伏のある形状の障害物との衝突を回避できた。

### 3.9 その他開発支援ツール

現場でのロボットのデバッグは、さまざまな条件の中で何が起きたかを素早く発見し、迅速に修正することが大切である。ロボットの状態確認としては、下記3項目を読み上げツールによってロボットにしゃべらせた。

- 次の目標の Waypoint の ID
- フラグ管理で正常でない組み合わせが発生したことの通知
- センサや中間ノードからトピックが来なくなったり時の通知

これにより、異常なゴール判定、想定できなかった状態遷移、各種機能の失敗をいち早く知ることができ、対策を本走行当日までに適用することができた。読み上げツールは OpenJTalk[12] を使用し、文字列のトピックを受け取るとそのまま読み上げるアプリケーションを作成し、ロボットに搭載されたディスプレイのスピーカーから音声を再生することで実現した。

## 4 性能評価

### 4.1 ハードウェア走行性能

走行ハードウェアとしては、CuboRex 製のクローラユニットである CuGo V3 を使用した。研究学園前公園の階段の乗降はできないが、必須課題の走行コースで問題になる段差はなかった。また、斜面のある通路、雨天時の走行においても、ほとんどクローラが滑ることなく走行、停止が可能だった。7/15 の実験走行では、炎天下の中必須課題の走行コースをマニュアル走行で4周したが、ハードウェアでのトラブルはなかった。

### 4.2 GNSS 位置推定

今回の手法を使用することで、GNSS 低精度エリア以外のコース区間では指定した通りの自律走行が実現できた。GNSS 低精度エリアを除けば、本走行、試走会ともに自己位置のロストを起因とする走行失敗・大きなルート逸脱は発生しなかった。しかし、同じ経路を指定して自律走行をさせた際に、最大 0.5[m] 程走行経路のずれが発生した。これは上位システムに起因する要素と、天気や湿度、準天頂衛星を含む測位衛星の位置といった、環境の変化が原因だと考えられる。つくばチャレンジの課題達成においては問題ない範囲の誤差であったが、狭路を含むルートの走行をこなすためには、3D Map Localization のような他の手法を併用する必要がある。

一方、GNSS 低精度エリアにおいては指示通りの自律走行は難しく、必ず市庁舎の壁から離れる方向に進んでしまった。これは、市庁舎のない方向からのマルチパスにより、自己位置が実際よりも市庁舎側に寄った位置にいると誤認してしまったためと考えられる。前述の通り、電波を遮蔽する物体が存在する環境においては本手法による位置推定は難しいため、ロボットの使用環境の調査と、遮蔽物が存在する場合は代替手法の検討が必須である。

### 4.3 2D 地図の位置推定



図 8 2D 地図の位置推定で発生した事象

図 5 に示す GNSS 低精度エリアとその前後は、AMCL による位置推定を使用している。図 7 で示した作成した地図では、下記 2 点の事象が発生した。

- 図 8 の位置 1 に商用バンが止まっていた時に著しく位置精度が低下した

- ・図 8 の位置 2 で 1 度だけ点対象の位置に誤マッチングしたことがあった。

いずれも確率は低かったが、あらゆるシーンで走行可能にするには、3D にすることやもっと遠くの地形を見ることなどの工夫が必要であると感じた。

#### 4.4 障害物回避



図 9 障害物回避のテスト

屋内・屋外のコンクリート床の上で下記 2 点試験が達成できるように、コストを参照する点の位置、障害物の有無を判定する閾値の調整した。

- ・走行可能部が機体幅とほぼ同等となる 1.2[m] の幅にパイロンを設置し、その間を自律走行で通過できる
- ・図 9 のように機体が直進走行中に機体の前方 0.5[m] の位置にパイロンを設置し、衝突することなく停止できる

### 5 本走行の結果

本走行では、確認走行区間を超えた先のパイロン地帯でパイロンを避けずに停止してしまったため走行を断念した。パイロンを避けきれなかった原因としては、以下の原因が考えられる。

1. 経路計画の動作周期が遅かった
2. 障害物を発見した際に停止する仕様とした
3. Waypoint を疎に打ちすぎていた
4. 当日の GNSS の値が最大 0.5m ほどずれていた。

1. は、経路計画の動作周波数が 1Hz だったため、パイロンを回避する経路が引かれる前に、3.8 に記載したロボットの停止が発生する位置までロボットが走行を続けたと考えられる。もとより、GNSS のみでどこまで走行できるかを主眼として開発を進めていた時期があり、GNSS と 2D 地図の位置推定が同居することを想定していなかった。本番直前にシステムを大幅に肥大化させたことにより、最適化が

進まず計算資源不足に陥ってしまったため、さまざまな制御周期を下げる対応をした。これにより、経路計画の動作周波数が 1Hz となってしまい、回避行動をとる前に障害物を認識する前の行動を続けてしまった。また、周囲の障害物を避けるコストマップの範囲も大幅に小さく設定したため、ロボットが障害物を発見するのにより障害物に近づいた状態でないと障害物に気づけない状況になっていた。

2. に関して、ロボットが他のロボットなどの障害物を回避する手段として、積極的に回避することをせずにその場でとどまるを選択してしまった。回避行動をとっていればとどまることはないが、回避行動をとることができず、この仕様によってデッドロックしてしまった。

3. に関して、確認走行区間では周囲のロボット以外の障害物は明確に知ることができており、Waypoint を 1.5m おきに密に設置し走行経路を明確に定義することで安定した走行を実現していた。しかし、パイロン区間は日によってパイロンの設置位置が変わるために、Waypoint の密度を 10m 以上の疎な設定していた。これによりパイロンから積極的に離れる経路にはならず、繰り返しパイロンに向かって走行する挙動をしてしまった。

4. に関して、GNSS の値は以前より比較的高精度に位置推定ができるようになったが、4.2 に記載した通り、位置推定の誤差が大きくなる場合が存在する。ロボットが停止してしまった区間は理想的なオープンスカイな環境ではあったが、もとから設定した Waypoint の位置より 0.5m 程度北にずれた位置を走行していた。パイロン地帯でも比較的設置されにくい点字ブロックに沿って走行をする経路を設定したが、そこから北にずれた経路を走行していたため積極的にパイロンが多い経路を走行し、パイロンを避けきれないケースが発生した。

パイロンを避けられなかった直接的な原因は以上の 4 点が挙げられるが、確認走行区間の先の対応をしたのが実験走行の最終日の午後のみで試行回数が圧倒的に少なかった。この先のほとんどの区間がオープンスカイな理想的な環境であり、運用次第では 2DLiDAR+GNSS+ホイールオドメトリのみという非常にシンプルな構成でもっと距離を伸ばせたと考える。この先の区間でしか現れない問題を収集できなかった点が今回の課題である。

## 6 結言

本年度のつくばチャレンジの取り組みとして、つくばの市街地で走行できるロボットハードウェアを作成し、GNSS と 2D 地図の位置推定を任意のタイミングで切り替えることができるナビゲーションシステムを作成し検討した。

走行実験では、確認走行区間の走行を実験走行最終日の検定と本走行を連続で達成した。しかし、その先の走行区間の調整や設定まで至れなかったため、パイロン区間で停止した。シンプルな構成で自律走行するシステムを実現したが、計算能力の限界や動的物体の障害物回避などに課題が残る。そのため、多くの現場で走行できるシステムにするためには最適化や計算資源の見直し、より広い範囲を認識することができるセンサ追加など、必要に応じて資源を増設できるシステムになるよう開発を続けたい。

## 参考文献

- [1] Geek+. Eve シリーズ, 2024. <https://www.geekplus.jp/product-2/poppick/?lan=JP>.
- [2] LexxPluss. hybrid-amr, 2024. <https://lexxpluss.com/hybrid-amr/>.
- [3] SEQSENSE. 次世代警備をリードする 自律移動型警備ロボット sq-2, 2024. <https://lexxpluss.com/hybrid-amr/>.
- [4] ugo. 警備 dx ソリューション, 2024. <https://ugo.plus/security-dx-solution/>.
- [5] CuboRex. E-cat kit2, 2023. <https://cuborex.com/product/?id=6>.
- [6] CuboRex. Cugo v4, 2023. <https://cuborex.com/product/?id=9>.
- [7] CuboRex. 不整地向けのロボット開発を加速！汎用クローラユニット「cugo」シリーズ最新作 ros 開発キットを販売開始, 2023. <https://prtentimes.jp/main/html/rd/p/000000057.000022568.html>.
- [8] みちびき. センチメータ級測位補強サービス, 2024. [https://qzss.go.jp/overview/services/sv06\\_clas.html](https://qzss.go.jp/overview/services/sv06_clas.html).