自動収穫ロボットの開発①
～SLAM を用いた自律移動の検証～

福島県立テクノアカデミー浜　職業能力開発短期大学校　計測制御工学科
○播磨　大希　　指導教官　牛坂　慶太

1. はじめに

近年、ロボットは工業用ロボットだけでなく、農業分野での開発が進められている。福島県では、農業ロボットを活用する農家も増えており、ロボットが注目されている。

本研究では、周辺環境の把握から地図作成が可能である、昨年度取り組んだ　SLAM（Simultaneous Localization And Mapping）に加えて、それを利用したナビゲーションを用いて、畑やビニールハウス内を自律的に運用できるロボットの開発を目的とした。

図 1　ひばり菜園ビニールハウス内

2. システム概要

(1) ROS

ROS（Robot Operating System）はオープンソースのミドルウェアの一つでソフトウェア開発者のロボット・アプリケーション作成を支援するライブラリとツールを提供している。 具体的には、ハードウェア抽象化、デバイスドライバ、ライブラリ、視覚化ツール、メッセージ通信、パッケージ管理などが提供されている。

(2)　SLAM

SLAM とは、(Simultaneously Localization and Mapping)の略で、各種センサから取得した情報をもとに、自己位置推定と地図製作を同時に行うことである。自立移動ロボットなどに利用される。

(3)　ナビゲーション

ナビゲーションとは、特定の環境でロボットをある場所から目的地まで移動するために、特定環境のオブジェクト、および壁のジオメトリ情報を含むマップが必要であり、その周辺情報を記憶することによって、制御通りの移動を行うことである。

具体的な動作は以下の通り

①事前推定
- 動作モデルに従ってサンプリング
- 観測に従いサンプル更新←作成途中の地図利用

②観測更新１
- 重みを計算←作成途中の地図と事前分布を利用

③地図更新
- 地図を更新する←事前推定による状態値を利用

④観測更新２
- 重みが最大となる粒子の状態値と地図を推定値とする
- 必要ならサンプリンクする。

⑤自己推定
- 作成した地図をもとに Rviz でロボットの位置を調整する。

⑥指定移動と方向設定
- Rviz で目標場所と方向を指定する。

(4)　環境構築

以下の項目を PC にセットアップ。

・RemotePC Setup
Ubuntu 16.04
ROS Kinetic Kame

・SBC(Raspberry PI3) Setup
Ubuntu　MATE　16.04

(5)　ロボット仕様

ロボットは昨年度「閉鎖空間における SLAM を用いた Mapping の検証」で使用したものと同様。

3. 実験

(1)　シミュレーションで動作確認

シミュレーションソフトである GAZEBO で使用して実機による検証の前にシミュレーションを行った。

・turtlebot3 を呼び出す

```
$ export=TURTLEBOT3_MODEL=burger
$ roslaunch turtlebot3_gazebo multi_turtlebot3.launch
```

・SLAM を実行

```
$ export=TURTLEBOT3_MODEL=burger
$ ROS_NAMESPACE=tb3_0 roslaunch turtlebot3_slam turtlebot3_gmapping.launch set_base_frame:=tb3_0/base_footprint set_odom_frame:=tb3_0/odom set_map_frame:=tb3_0/map
```

・turtlebot3 を地図データと統合

```
$ sudo apt-get install ros-kinetic-multirobot-map-merge
$ roslaunch turtlebot3_gazebo multi_map_merge.launch
```

・Rviz を実行

$ rosrun rviz rviz -d `rospack find turtlebot3_gazebo`/rviz/multi_turtlebot3_slam.rviz

・遠隔操作

$ export=TURTLEBOT3_MODEL=burger

$ ROS_NAMESPACE=tb3_0 rosrun turtlebot3_teleop turtlebot3_teleop_key

・地図を保存

$ rosrun map_server map_saver -f ~/map

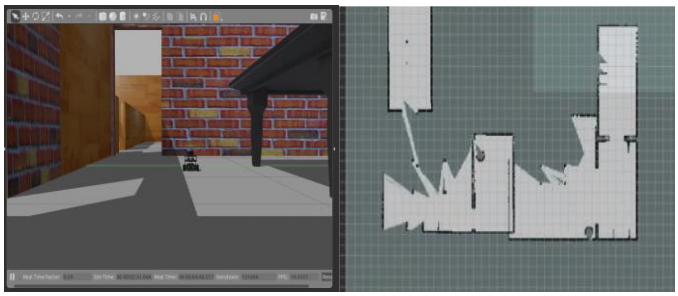

図 2　シミュレーション内での Mapping

(2)　Remote PC と TurtleBot3 通信設定

互いの IP アドレスを設定する。

・ROS_MASTER_URI=http://192.168.17.160:11311

・ROS_HOSTNAME=192.168.17.166

(3)　地図生成

地図生成パッケージは複数存在するが、今回は一般的に SLAM で使用される Google　mapping を使用した。制作したコースを遠隔操作しているロボットで地図作成を行った結果、図 3 の結果が得られた。

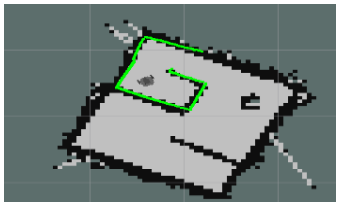

図 3　Google　Mapping

本ロボットは、レーザーセンサで物体までの距離を測る事が可能であり、前述した Google mapping の地図を元に精度の検証を行った。

(4)　ナビゲーションノードを実行

$ roscore

$ roslaunch turtlebot3_bringup turtlebot3_robot.launch

$ export TURTLEBOT3_MODEL=burger

$ roslaunch turtlebot3_navigationturtlebot3_navigation.launch map_file:=$HOME/map.yaml

$ rviz –d `rospackfind turtlebot3_navigation`/rviz/turtlebot3_navigation.rvi

図 4　ナビゲーションノードを実行

(5)　ロボットの初期姿勢を設定

Rviz のメニューで 2D Pose Estimate でドラッグすると図 4 のように大きな緑色の矢印が表示されるのでロボットが向いている方向にドラッグする。

次に、keyboard ノードなどのツールを使用してロボットを前後に移動して、周囲の環境情報を収集し、ロボットが現在マップ上のどこにあるかを調べる。

$ export TURTLEBOT3_MODEL=burger

$ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch

(6)　ナビゲーションの目標を送信

ナビゲーションの目標を送信すると、ロボットはパスに沿って移動する。このとき、障害物が突然検出されても、ロボットは LIDAR からの情報によって障害物を避けて目標点に移動する。検証の結果を以下に示す。

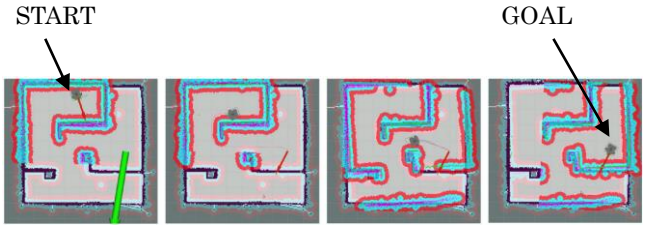

図 5　ナビゲーションの結果

図 5 に示す通りロボットの実際の位置と作成した地図上での位置が同様であることがわかり、ナビゲーションによる自己位置の推定と遠隔操作が正確に行われていることが確認できる。

4. 結果

ロボットを遠隔操作し Google Mapping を用いた SLAM&ナビゲーションによって未知なる環境でも正確に地図作成と共に自己位置の推定、遠隔操作を行える事が分かった。

5. おわりに

今回 SLAM&ナビゲーションを用いた自律移動の検証を行った。実験を通して SLAM&ナビゲーションについてより詳しく知ることができたが、改善点も多く見つかった。今後の課題としては、ナビゲーションで畑やビニールハウス内での正確な運用ができるよう取り組んでいきたい。

参考文献

(1)　ROBOTIS e-Manual　http://emanual.robotis.com/

(2)　SLAM について産業技術総合研究所知能システム研究部門