

User's Documents for ADAMR2 Project

一関高専 専攻科 生産工学専攻 2年 藤野航汰

2020年10月30日

目次

第 1 章	プロジェクト概要	2
1.1	背景	2
1.2	自律移動ロボットプラットフォーム ADAMR2	2
1.3	動作環境	2
第 2 章	セットアップガイド	4
2.1	電装の繋ぎ方	4
2.2	部品一覧	4
2.3	接続図	5
2.4	端子の接続方法の詳細	5
2.5	ROS Melodic のインストール	7
2.6	ADAMR2 用 ROS パッケージのインストール	8

第1章

プロジェクト概要

1.1 背景

社会実装指向型研究では、研究成果を現場へ持ち込み、ユーザのレビューから改善点を即座にフィードバックするアジャイル型開発のアプローチが要求されます。しかし、ロボット等のメカニカルデバイスを必要とする研究テーマでは、デバイスの設計開発に多くの時間を必要とすることから、アジャイル型開発が適用しにくい状況がありました。

この課題に対し、一関高専では、東京高専・和歌山高専と共同で自律移動ロボットプラットフォームの開発に取り組んでいます。これは、自律移動に関する部分をプラットフォーム化して提供することで、目的の機能の開発に注力できることを目標としたものです。これまでの研究で、図 1.1 に示すような大径インホイールモータを搭載した移動ロボットが開発されました。

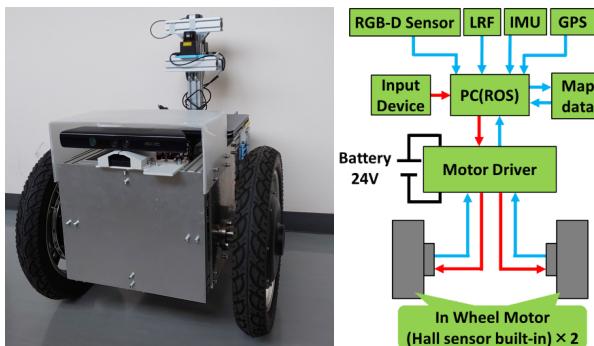


Fig.1.1 Appearance and System Configuration of The Mobile Robot Platform Developed so far

1.2 自律移動ロボットプラットフォーム ADAMR2

現在、図 1.2 に示すような対向 2 輪型移動ロボット「ADAMR2」が開発されています。アジャイル型開発に対応できるという意味（というよりも願い）を込めて、「Agile Developable Autonomous Mobile Robot ver.2」と名付けました。

1.3 動作環境

ADAMR2 は ROS^{*1}を活用して開発されています。ROS パッケージは GitHub 上で公開されており、誰でも利用することができます。

ADAMR2 の ROS パッケージは以下の環境で動作することを確認しています。

- Intel NUC8i7BEH
- CPU: 8th Gen quad-core intel ® Core™ i7 Processor
- GPU: Iris ® Plus Graphics 655
- RAM: Crucial 16GB DDR4-2400 SODIMM CT16G4SFD824A

*1 <http://wiki.ros.org/>

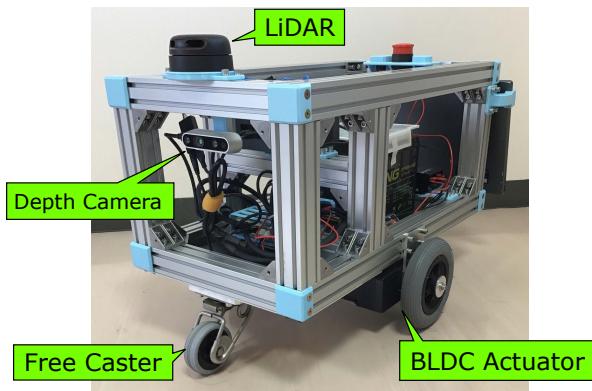


Fig.1.2 Appearance of ADAMR2

- Ubuntu 18.04 LTS
- ROS Melodic Morenia

第2章

セットアップガイド

2.1 電装の繋ぎ方

鉛蓄電池、スイッチ、モータードライバ等のロボットの電装を接続します。

2.2 部品一覧

電装の配線には以下の部品を使用します。

2.2.1 電装部品一覧

Name	Model Number	Quantity
鉛蓄電池 12V 20Ah	WP20-12IE	2
端子台	ATK-30-6P	1
メイン電源用スイッチ	273-025	1
DC コンバータ用スイッチ	RB625011FF-10P	1
DC コンバータ 12V-19V	a16052500ux0037	1
DC プラグ	761KS15	1
T-Frog モータードライバ	TF-2MD3-R6	1
T-Frog 電源基板	TF-PW36-5/12M	1

Table2.1 List of Electrical Components

2.2.2 配線一覧

Name	Model Number
ニチフ 銅線用絶縁被覆付圧着端子丸型	TMEV1.25-3
差込み型接続端子 187 シリーズ (バリュー品) メス (嵌合部絶縁型)	MTR-480809-FA
絶縁付圧着端子 Y 型	F1.25-5
通信機器用ビニル電線 KV シリーズ	KV 1.25SQ Φ6-200
通信機器用ビニル電線 KV シリーズ	KV 1.25SQ Φ6-200

Table2.2 List of Wiring Components

2.3 接続図

電装部品の接続図を図 2.1 に示します。

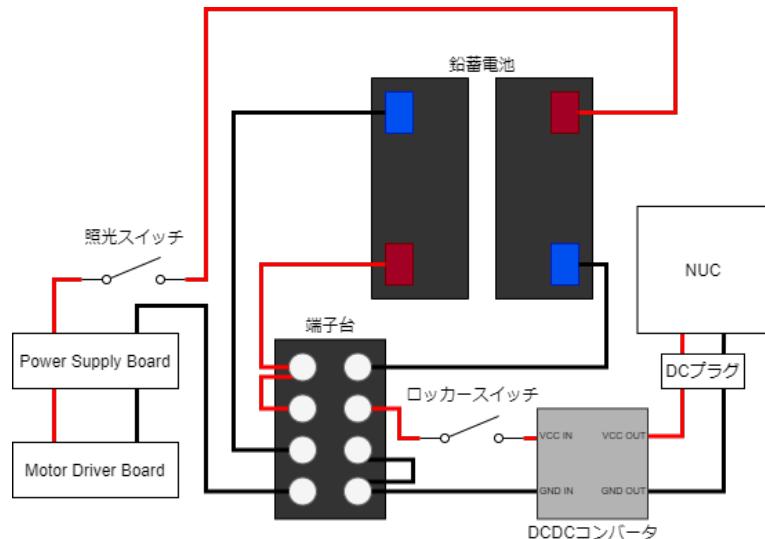


Fig.2.1 Electrical Connection Diagram

2.4 端子の接続方法の詳細

2.4.1 鉛蓄電池

鉛蓄電池 WP20-12IE は M5 ねじの接続端子を持っているので、Y 型圧着端子をねじ止めすることで接続することができます。ねじの締結には 8mm 幅のレンチを使用できます。感電やショートに十分注意して作業してください。



Fig.2.2 Detail of How to Connect Pb-Battery

2.4.2 端子台

端子台 ATK-30-6P は M5 ねじの接続端子を持っているので、Y 型圧着端子をねじ止めすることで接続できます。

1 つの端子に 2 つの圧着端子を接続することもできるので、圧着端子を両端につけたケーブルを使って 2 つの端子列をショートさせることができます。上の接続例でも、GND や 12V の箇所でこのような接続方法をとっています。



Fig.2.3 Detail of How to Connect Screw Terminal Block

2.4.3 スイッチ

ADAMR2 で使用しているスイッチは、いずれも 187 シリーズの差込型接続端子（ファストン端子）を持っています。接続には 187 シリーズの差込型端子（メス）を付けたケーブルを使用する必要があります。

差込型接続端子ははんだ付け無しで何度も取り外しができますが、付け直す度に端子が摩擦で削れて接続が緩くなる可能性があります。取り外しはできるだけ最小限に抑えるよう注意してください。

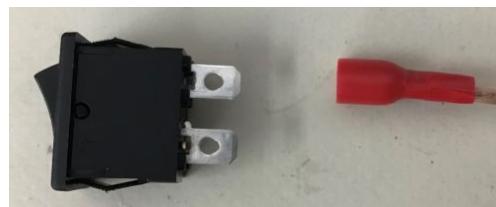


Fig.2.4 Detail of How to Connect Switch

2.4.4 T-Frog 電源基板

T-Frog 電源基板は、M3 ねじ止めターミナルを持っていて、接続には内径Φ 3.2 の圧着端子を使用します。ADAMR2 では丸形圧着端子を使用しています。

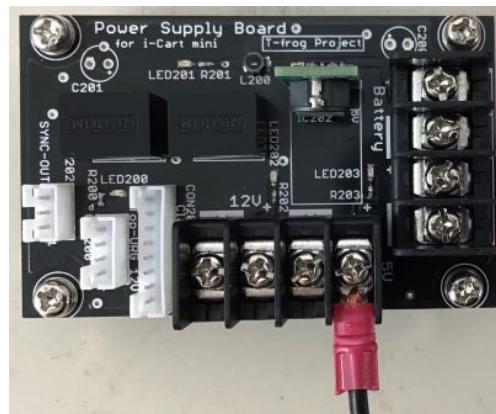


Fig.2.5 Detail of How to Connect T-Frog Power Supply Board

2.5 ROS Melodic のインストール

ROS Melodic Morenia をインストールします。インストール手順は ROS Wiki^{*1}に書かれている内容と同じですが、ビルドシステムに catkin ビルドツールを使用する点が異なります。

2.5.1 ROS 本体のインストール

Ctrl + Alt + T を押してターミナルを開き、以下のコマンドを順番に実行します。

【注意】PDF のコードブロックはコピーペーストした際に文章が崩れる可能性があります。コマンドのコピペは ROS Wiki から行うことをお勧めします。

Code 2.1 リポジトリの鍵の取得

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" \  
> /etc/apt/sources.list.d/ros-latest.list'
```

Code 2.2 鍵の登録

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key \  
C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

Code 2.3 パッケージリストのアップデート

```
sudo apt update
```

ここではデスクトップ版の ROS をインストールします。

Code 2.4 ROS 本体のインストール

```
sudo apt install ros-melodic-desktop-full
```

Code 2.5 ROS のセットアップスクリプトの読み込みを自動化する

```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc  
source ~/.bashrc
```

Code 2.6 ユーティリティツールのインストール

```
sudo apt install \  
python-rosdep \  
python-rosinstall \  
python-rosinstall-generator \  
python-wstool \  
build-essential
```

Code 2.7 rosdep のインストール

```
sudo rosdep init  
rosdep update
```

2.5.2 catkin ビルドツールのインストール

ROS1 のモダンなビルドツールとして、Catkin Command Line Tools^{*2}が開発されています。

*1 <http://wiki.ros.org/melodic/Installation/Ubuntu>

*2 <https://catkin-tools.readthedocs.io/en/latest/>

ROS Wiki や ROS の教本などでは `catkin_make` コマンドを使ってワークスペースの設定やパッケージのビルドを行っていますが、`catkin` ビルドツールの方が使い勝手が良いです。これから ROS1 を始める際は `catkin_make` コマンドでなくこちらを使うことをおすすめします。

Catkin Command Line Tools はデフォルトで ROS に含まれていないので、追加でインストールする必要があります。

Code 2.8 `catkin` ビルドツールのインストール

```
sudo apt install python-catkin-tools
```

【注意】`catkin` コマンドと `catkin_make` コマンドの間には互換性がありません。そのため、`catkin_make` コマンドで初期化したワークスペースを `catkin` コマンドでビルドすることはできません。もし既に `catkin_make` コマンドでワークスペースを初期化している場合には、`catkin_ws` ディレクトリにある `src/`以外のディレクトリを削除してから `catkin` コマンドでビルドし直してください。

2.5.3 ワークスペースの初期化

`catkin` コマンドを使って、ROS のワークスペースを初期化します。

Code 2.9 ワークスペースの初期化

```
mkdir -p ~/catkin_ws/src  
cd ~/catkin_ws  
catkin init
```

これで ROS のインストール作業は終了です。次に、ADAMR2 用の ROS パッケージをインストールする作業を行います。

2.6 ADAMR2 用 ROS パッケージのインストール

ADAMR2 用の ROS パッケージをインストールします。パッケージをダウンロードする際に Git^{*3}が必要になります。git がインストールされていない場合は、追加でインストールしましょう。

Code 2.10 Git のインストール

```
sudo apt install git
```

2.6.1 パッケージのクローン

GitHub^{*4} から ADAMR2 用 ROS パッケージをワークスペースにクローンします。melodic-devel ブランチをクローンしましょう。

Code 2.11 パッケージのクローン

```
cd ~/catkin_ws/src  
git clone -b melodic-devel https://github.com/fjnkt98/adamr2_ros1_packages.git
```

この時点ではまだビルド時依存パッケージである YP-Spur^{*5}がインストールされていないので、パッケージのビルドはできません。YP-Spur のインストールは次の小節で行います。

2.6.2 依存パッケージのインストール

ADAMR2 用 ROS パッケージが依存しているパッケージを `rosdep` を用いてインストールします。コード 2.12 のコマンドを実行します。

*3 <https://git-scm.com/>

*4 https://github.com/fjnkt98/adamr2_ros1_packages/tree/melodic-devel

*5 <https://openspur.org/>

Code 2.12 依存パッケージのインストール

```
rosdep install -i --from-path ~/catkin_ws/src/adamr2_ros1_packages/
```

このコマンドを実行することで、rosdepが依存関係を解決し、不足しているパッケージをapt-getコマンドを使用して自動的にインストールしてくれます。

この作業を行うことで、YP-Spurのバイナリも一緒にインストールされます。また、YP-SpurのROS ラッパーとしてypspur_ros^{*6}パッケージも提供されていますが、ADAMR2では使用しません。

2.6.3 パッケージのビルド

依存関係を解決したので、パッケージをビルドできるようになりました。コード2.13のコマンドを実行してパッケージをビルドします。

Code 2.13 パッケージのビルド

```
cd ~/catkin_ws  
catkin build
```

ビルドが完了したら、ワークスペース内のsetup.bashスクリプトを読み込みます。

Code 2.14 セットアップスクリプトの読み込み

```
source ~/catkin_ws/devel/setup.bash
```

ついでに~/.bashrcに書き込んで自動化してしまいましょう。

Code 2.15 セットアップスクリプト読み込みの自動化

```
echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
```

2.6.4 環境設定

実際にロボットの実機を動かす際は、いくつかの環境設定を行う必要があります。

まずは実行ファイルへの権限付与と実験用ディレクトリの生成を行います。adamr2_ros1_packages/utilityディレクトリ内にあるsetup_environment.shを実行します。

Code 2.16 環境設定スクリプトの実行

```
cd ~/catkin_ws/src/adamr2_ros1_packages/utility  
bash setup_environment.sh
```

このシェルスクリプトは以下の処理を実行しています。

- ypspur-coordinatorを起動するスクリプトypspur_launcher.shに実行権限を付与する
- 実験で取得したrosbagファイルや地図ファイルを保存しておくためのディレクトリをホームディレクトリに生成する

手動で行うのが面倒な作業を一括で行ってくれます。

次に、USBデバイス名の固定を行います。T-Frogモータードライバ及びRPLiDARをPCに接続した状態で、adamr2_ros1_packages/utility/udevディレクトリ内にあるset_udev_rules.shを実行します。

Code 2.17 udev rulesの設定

```
cd ~/catkin_ws/src/adamr2_ros1_packages/utility  
bash set_udev_rules.sh
```

内部でsudoコマンドを実行しているため、このシェルスクリプトを実行する際はパスワードが要求されます。

*6 https://github.com/openspur/ypspur_ros

このシェルスクリプトを実行することによって、T-Frog モータードライバと RPLiDAR のデバイスパスのシンボリックリンク /dev/tfrog-driver と /dev/rplidar が生成されます。

ここまで手順により、ROS の開発環境を整えることができました。ハードウェアのセットアップが完了しているなら、各パッケージにある launch ファイルを実行することでロボットを動かすことができます。