

# 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 12 月 7 日

# 目次

<b>第 1 章</b>	<b>プロジェクト概要</b>	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
<b>第 2 章</b>	<b>セットアップガイド -ハードウェア編-</b>	4
2.1	電装の繋ぎ方	4
2.2	USB デバイスの接続	7
<b>第 3 章</b>	<b>セットアップガイド -ソフトウェア編-</b>	9
3.1	Ubuntu 18.04LTS のインストール	9
3.2	ROS Melodic のインストール	10
3.3	ADAMR2 用 ROS パッケージのインストール	12

# 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。

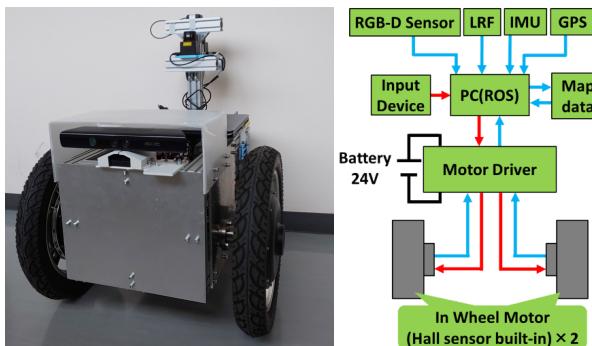


Fig.1.1 Appearance and System Configuration of The Mobile Robot Platform Developed so far

### 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<sup>\*1</sup>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/>

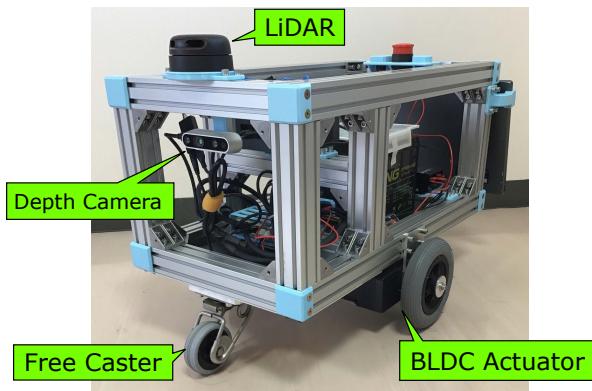


Fig.1.2 Appearance of ADAMR2

- Ubuntu 18.04 LTS
- ROS Melodic Morenia

## 第2章

# セットアップガイド -ハードウェア編-

### 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。電装の接続には主に絶縁被覆付圧着端子を使用します。

#### 2.1.1 部品一覧

電装の配線には以下の部品を使用します。

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
緊急停止スイッチ基板	-	1

Table2.1 List of Electrical Components

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 Α-200
通信機器用ビニル電線 KV シリーズ	KV 1.25SQ Κ-200

Table2.2 List of Wiring Components

#### 2.1.2 必要となる工具

電装の取り付け作業には以下の工具が必要となります。

- 二面幅 8 mm のレンチ
- プラスドライバ (No.2 以上)
- 絶縁被覆付圧着端子用圧着工具 ホーザン P-743

### ● ワイヤーストリッパ

鉛蓄電池の端子は二面幅 8 mm の六角ナット / ボルトが使用されているので、二面幅 8 mm のレンチが必要になります。電池を取り扱うので、絶縁されている工具が望ましいです。また、レンチにラチェット機能が付いていると作業性が非常に向上します。ラチェット機能付きのソケットレンチを使用することをお勧めします。

ADAMR2 の電装で使用している圧着端子はほぼ全て絶縁被覆なので、圧着工具も絶縁被覆付圧着端子用のものを使用する必要があります。裸圧着端子用の工具を使用すると被覆が破れて不具合が発生する恐れがあるので、使用しないでください。

### 2.1.3 接続図

電装部品の接続図を図 2.1 に示します。

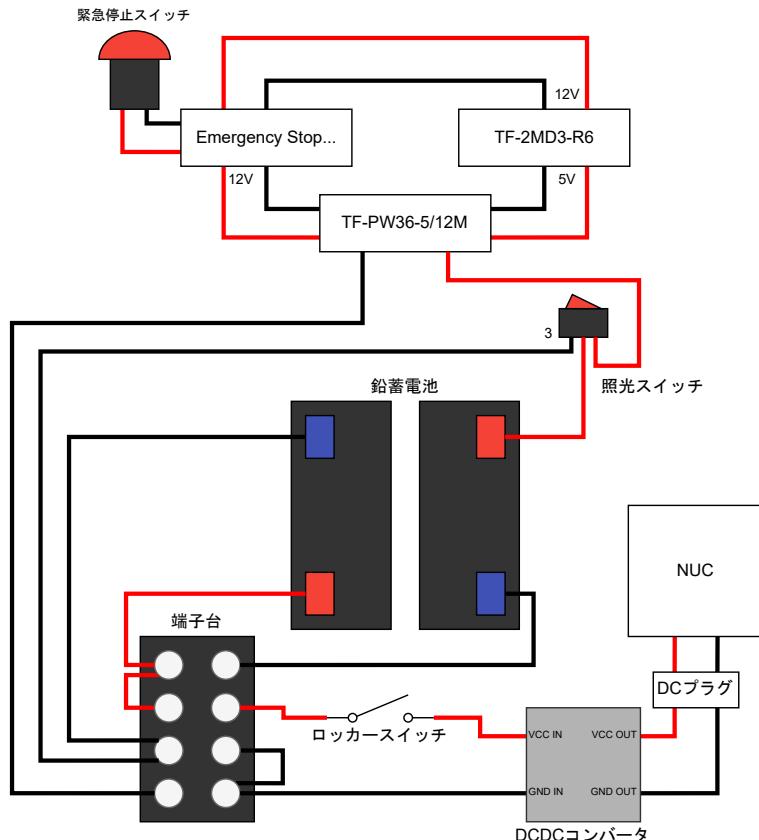


Fig.2.1 Electrical Connection Diagram

### 2.1.4 端子の接続方法の詳細

#### 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。

#### 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.2.2 Detail of How to Connect Pb-Battery

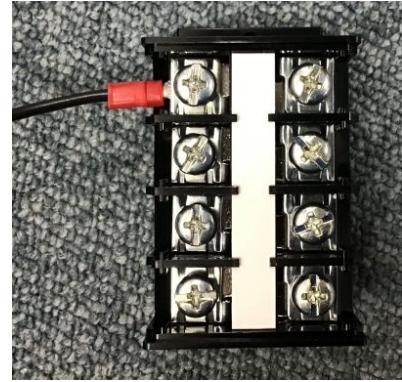


Fig.2.3 Detail of How to Connect Screw Terminal Block

### 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。

### 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  $\phi$  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。

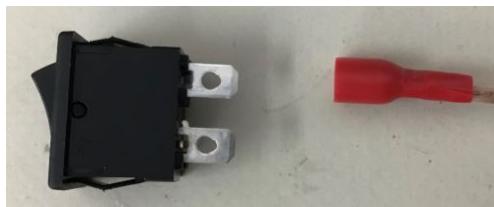


Fig.2.4 Detail of How to Connect Switch

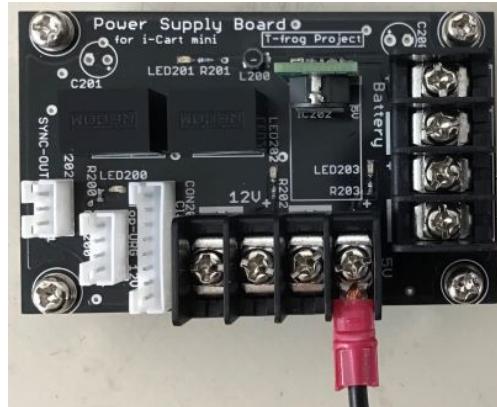


Fig.2.5 Detail of How to Connect T-Frog Power Supply Board

### T-Frog モータードライバ

T-Frog モータードライバは、専用のコネクタを介してロジック電源・モーター電源の 2 種類の電源を接続します。購入時に付属してくるコネクタをケーブルに取り付けて接続します。詳細は i-cart mini 組立説明書 ([http://t-frog.com/products/icart\\_mini/assembly/](http://t-frog.com/products/icart_mini/assembly/)) を参照してください。

### 緊急停止スイッチ基板

緊急停止スイッチ基板は ADAMR2 プロジェクトオリジナルの基板です。外観を図 2.6 に示します。パワー リレーと緊急停止スイッチを用いて、モーター電源を遮断できるようになっています。T-Frog 電源基板に使用されているものと同一のねじ止めターミナルを持っており、モーター電源はここに接続します。緊急停止スイッチは XH コネクタによって接続します。

この緊急停止スイッチ基板の設計データは [https://github.com/fjnkt98/adamr2\\_emg\\_board/releases/tag/v0.1.0](https://github.com/fjnkt98/adamr2_emg_board/releases/tag/v0.1.0) にて公開されているため、誰でも同じものを作ることができます。



Fig.2.6 Emergency Stop Switch Circuit Board for ADAMR2

## 2.2 USB デバイスの接続

ADAMR2 では以下の USB デバイスを使用します：

- T-Frog モータードライバ (TF-2MD3-R6)
- RPLiDAR A2
- Intel Realsense D435
- Logicoool F710 ゲームパッド (オプション)

他に、ディスプレイとして HDMI 接続のモバイルディスプレイを接続し、モバイルディスプレイと RPLiDAR に 5V 電源を供給するためのモバイルバッテリーも搭載します。

各種デバイスの接続図を図 2.7 に示します。

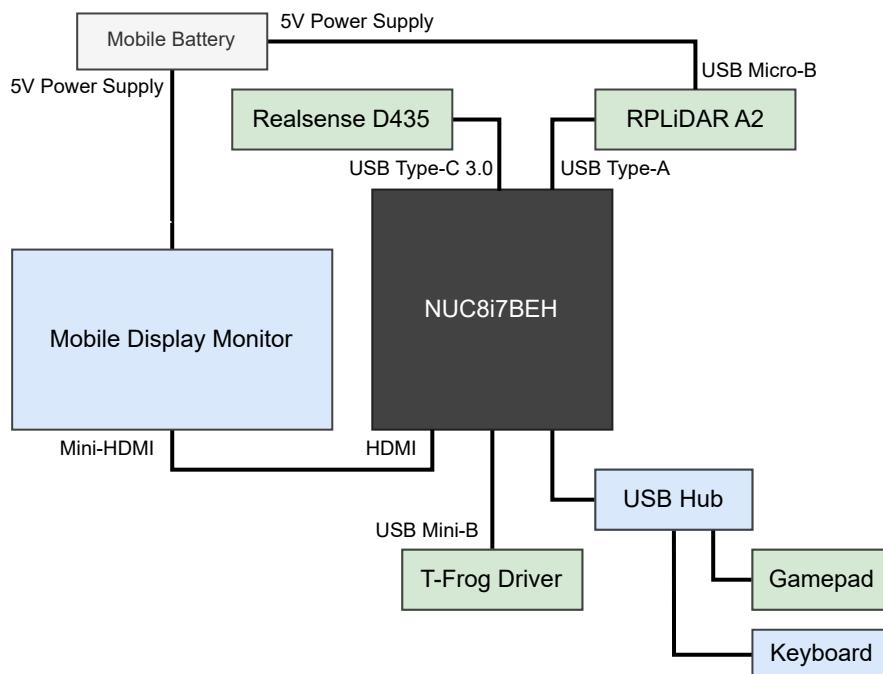


Fig.2.7 USB Device Connection Diagram

注意点として、Intel Realsense D435 と RPLiDAR A2 は、USB ハブを介した接続を行うと電流不足で動

作しなくなる場合があります。PC の USB ポートに直接接続し、電流不足を回避してください。<sup>\*1</sup>

---

<sup>\*1</sup> RPLiDAR は外部給電を行っているため電流不足にはならないはずなのですが、実際に電流不足に陥ることが多かったため、可能な限り PC の USB ポートに直接接続するようにしてください。

## 第3章

# セットアップガイド－ソフトウェア編－

### 3.1 Ubuntu 18.04LTS のインストール

この節では Intel NUC へ Ubuntu 18.04LTS をインストールする手順を説明します。尚、NUC の組み立ては完了しているものとします。

#### 3.1.1 必要なもののリスト

Ubuntu のインストール作業には以下のものが必要となります。

- USB メモリ (8GB 以上が望ましい)
- HDMI 入力端子付きディスプレイ
- HDMI ケーブル
- キーボード・マウス (有線)

尚、Wi-Fi 環境がある場合は LAN ケーブル及び有線 LAN 環境は必ずしも必要ではありません。

#### 3.1.2 ライブ USB の作成

NUC へ Ubuntu をインストールするには、Ubuntu OS を記録したメディアを作成する必要があります。今回は USB メモリへ Ubuntu OS の ISO ファイルを書き込んでライブ USB を作成します。

ライブ USB の作成は Windows10 でも行えます。まずは Ubuntu 18.04LTS の ISO イメージファイルと、ライブ USB 作成ソフトの Rufus<sup>\*1</sup>をダウンロードします。

Ubuntu 18.04LTS の ISO ファイルは、以下のミラーサーバからダウンロードすることができます。どのサーバからダウンロードしても構いません。

- 富士大学: <http://cdimage-u-toyama.ubuntulinux.jp/releases/18.04.3/>
- JAIST: <http://ftp.jaist.ac.jp/pub/Linux/ubuntu-jp-cdimage/releases/18.04.3/>
- KDDI 研究所: <http://ftp.kddilabs.jp/Linux/packages/ubuntu-jp/release-cd/releases/18.04.3/>
- 株式会社アプセル: <http://cdimage-appcel.ubuntulinux.jp/releases/18.04.3/>

ダウンロードするファイルは、ubuntu-ja-18.04.3-desktop-amd64.iso です。ファイルサイズが 1.9 GB のものをダウンロードします。

次に Rufus をダウンロードします。公式サイト <https://rufus.ie/> から最新版をダウンロードします。インストールは不要で、.exe ファイルがダウンロードされるのでそれを起動します。

Rufus を起動すると図 3.1.2 のようなダイアログが表示されます。

「デバイス」の欄にはライブ USB にする USB メモリを選択します。「ブートの種類」の欄には先ほどダウンロードした Ubuntu 18.04LTS の ISO イメージファイルを指定します。

「スタート」ボタンをクリックすれば書き込みが開始されます。イメージを書き込んだ USB メモリの中身は全て消去されます。重要なファイルが入っている場合は事前にバックアップを取っておきましょう。

無事にライブ USB が作成できたら、いよいよ Ubuntu のインストール作業に入ります。

---

<sup>\*1</sup> Fedora Media Writer でもライブ USB の作成を行えます。好きな方を選んでください。

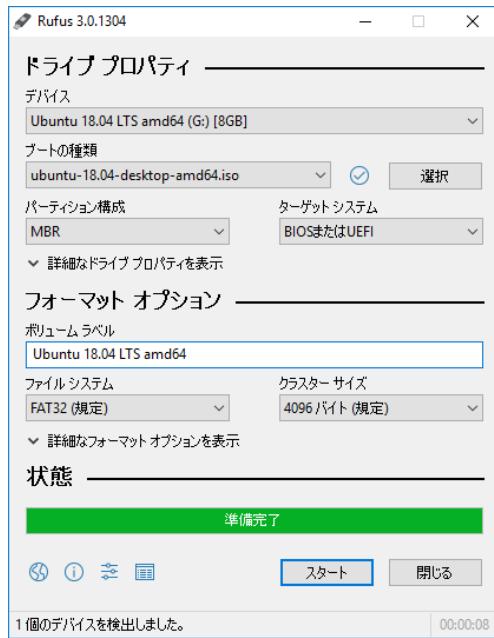


Fig.3.1 Dialog of Rufus

### 3.1.3 Ubuntu のインストール

作成したライブUSBをNUCに接続し、ディスプレイやキーボード等を繋いでNUCを起動します。「Install Ubuntu」を選択して、Ubuntuのインストールを開始します。Ubuntuのインストールはウィザードの指示に従って作業を行えば大丈夫です。

## 3.2 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<sup>\*2</sup>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。

### 3.2.1 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 3.1 リポジトリの鍵の取得

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

Code 3.2 鍵の登録

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

Code 3.3 パッケージリストのアップデート

```
sudo apt update
```

<sup>\*2</sup> <http://wiki.ros.org/melodic/Installation/Ubuntu>

ここではデスクトップ版の ROS をインストールします。

Code 3.4 ROS 本体のインストール

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

Code 3.5 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 3.6 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 3.7 `rosdep` のインストール

```
sudo rosdep init
rosdep update
```

### 3.2.2 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<sup>\*3</sup>が開発されています。

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 3.8 `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。

### 3.2.3 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 3.9 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。

\*3 <https://catkin-tools.readthedocs.io/en/latest/>

### 3.3 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<sup>\*4</sup>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 3.10 Git のインストール

```
sudo apt install git
```

#### 3.3.1 パッケージのクローン

GitHub<sup>\*5</sup>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 3.11 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<sup>\*6</sup>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。

#### 3.3.2 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 3.12 のコマンドを実行します。

Code 3.12 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<sup>\*7</sup> パッケージも提供されていますが、ADAMR2 では使用しません。

#### 3.3.3 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 3.13 のコマンドを実行してパッケージをビルドします。

Code 3.13 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 3.14 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。

<sup>\*4</sup> <https://git-scm.com/>

<sup>\*5</sup> [https://github.com/fjnkt98/adamr2\\_ros1\\_packages/tree/melodic-devel](https://github.com/fjnkt98/adamr2_ros1_packages/tree/melodic-devel)

<sup>\*6</sup> <https://openspur.org/>

<sup>\*7</sup> [https://github.com/openspur/ypspur\\_ros](https://github.com/openspur/ypspur_ros)

Code 3.15 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
```

### 3.3.4 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 3.16 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 3.17 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。

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 のデバイスバ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。