

20コマ目 ROSのプログラム（パッケージ） 実行



目次

1. ROSプログラム作成
2. ROSプログラム実行

ROSのプログラム作成



作成プログラム

- RobotCarの中でROSプログラムを2つ起動して，制御する



ファイルのダウンロード

- 以下のURLからファイルを受講者PCにダウンロードする

<https://rtc-fukushima.jp/wp/wp-content/uploads/2018/11/20ROS-2.zip>

- 以下のファイルがあることを確認してください。
 - CMakeList.txt
 - InPutVolte.py
 - OutPutVolte.py
 - 20ROS.txt
 - package.xml
 - 以降のコマンドはこのファイルからコピーすることが可能

作成手順

- ROSのプログラムを作成するに以下の手順を行う
 1. ROSの設定反映
 2. ROSのワークスペースを作成
 3. ROSのワークスペースをビルド1
 4. ワークスペースをインストール環境に上書き
 5. catkin形式のパッケージを作成
 6. package.xmlの修正
 7. ROSのワークスペースをビルド2
 8. メッセージの作成
 9. CMakeList.txtを変更
 10. ROSのワークスペースをビルド3
 11. プログラムを作成
 12. ROSのワークスペースをビルド4

ROSの設定反映

- ROS設定を反映させるために，bashを実行する

```
$ source /opt/ros/kinetic/setup.bash
```

- このbashを実行することによって，ROS特有のコマンドが実行可能になる

ROSのワークスペースを作成

- ROSのパッケージを作成する作業ディレクトリを作成する

```
$ mkdir -p ~/catkin_ws/src
```

```
$ cd ~/catkin_ws/src
```

```
$ catkin_init_workspace
```

- **catkin_init_workspace**

- ワークスペースを作成するコマンドであり、
このコマンドを実行したディレクトリ以下で作業をする

ROSのワークスペースをビルド1

- ROSのパッケージを作成する作業ビルドを作成する

```
$ cd ~/catkin_ws/
```

```
$ catkin_make
```

- **lsコマンド**で3つのディレクトリが作成されていることを確認
 - build：ビルドの設定
 - devel：実行ファイルで生成されたものはいっている
 - src：プログラムを保存する場所

ワークスペースをインストール環境に 上書き

- 以下のコマンドを入力する

```
$ source devel/setup.bash
```

- パスの確認

- Overlayが出来ているか確認

```
$ echo $ROS_PACKAGE_PATH
```

- 以下のようにワークスペースが入っていること確認
/home/pi/catkin_ws/src:/opt/ros/kinetic/share

catkin形式のパッケージを作成

- ROSのパッケージを作成

```
$ cd ~/catkin_ws/src
```

```
$ catkin_create_pkg robotcar_pkg std_msgs rospy roscpp
```

- **catkin_create_pkg [パッケージ名] [依存パッケージ1] [依存パッケージ2]...**

- 依存パッケージはそのパッケージを作成するのに必要なパッケージ

- **std_msgs** : メッセージ通信を行うパッケージ
 - **rospy** : Pythonでプログラムを行うパッケージ
 - **roscpp** : C++でプログラムを行うパッケージ

- lsコマンドで**[robotcar_pkg]**が作成されていることを確認

package.xmlの修正

- メッセージ通信を行うために、packge.xmlを修正する
- `<!-- <build_depend>message_generation</build_depend> -->` の`<!--` と `-->` (コメントタグ) を削除
- `<!-- <exec_depend>message_runtime</exec_depend> -->` の `<!--` と `-->` (コメントタグ) を削除
- この講習会は編集せずに，編集済みのファイルで上書きする
/home/pi/catkin_ws/src/robotcar_pkg内のpackage.xmlに上書き

ROSのワークスペースをビルド2

- ROSパッケージを作成したので，ワークスペースをビルドする

```
$ cd ~/catkin_ws/
```

```
$ catkin_make
```

メッセージの作成

- メッセージ通信時に，送信したい変数と型を決定

```
$ cd src/robotcar_pkg/
$ mkdir msg
$ echo int32 left_sval >> msg/Msg.msg
$ echo int32 right_sval >> msg/Msg.msg
$ rosmmsg show Msg
```

- [rosmmsg]実行時に以下の結果が出ることを確認

```
[robotcar_pkg/Msg]:
int32 left_sval
int32 right_sval
```

CMakeLists.txtを変更

- メッセージ通信を行うために，CMakeLists.txtを修正する
 - message_generation追加
 - find_package内にmessage_generationを追加する
 - add_message_files修正
 - add_message_filesのコメントアウト(#)削除
 - Message1.msg, Message2.msg削除
 - Msg.msg追加
 - generate_messages修正
 - generate_messagesのコメントアウト(#)削除
 - catkin_package修正
 - CATKIN_DEPENDSのコメントアウト(#)削除
- この講習会は編集せずに，編集済みのファイルで上書きする
 /home/pi/catkin_ws/src/robotcar_pkg内のCMakeLists.txtに上書き

ROSのワークスペースをビルド3

- メッセージを作成したので，ワークスペースをビルドする

```
$ touch CMakeLists.txt
```

```
$ cd ~/catkin_ws/
```

```
$ catkin_make
```


パッケージのプログラム部分を作成



プログラムを配置

- プログラムを格納するディレクトリを作成

```
$ roscd robotcar_pkg
```

```
$ mkdir scripts
```

```
$ cd scripts
```

- scripts内に以下のプログラムをコピーします。

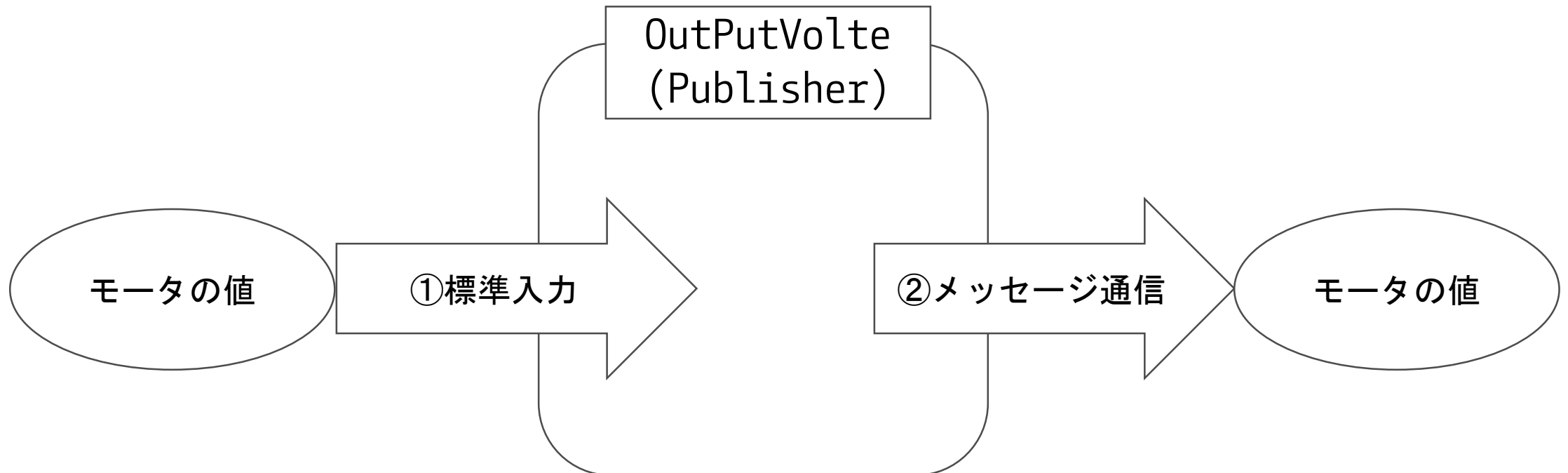
InPutVolte.py, OutPutVolte.py

- パーミッションを変更します。

```
$ chmod +x InPutVolte.py OutPutVolte.py
```

OutPutVolte(Publisher)

- プログラム概要
 - ① 標準入力でモータの値を入力
 - ② メッセージ通信で値を出力



OutPutVolte(Publisher)

- ROSのプログラミングに必要なimport
 - PythonでROSプログラムを作成
 - `import rospy # pythonでROSのプログラムを記述`
 - 作成したメッセージを使用する宣言
 - `from robotcar_pkg.msg import Msg # 作成したメッセージを使用する宣言`
- Try (例外処理) を使用するためにimport
 - `import sys # 例外処理`

OutPutVolte(Publisher)

- メインメソッド
 - プログラムの内容は以下のメソッドに書かれる

```
if __name__ == '__main__':
```

- Ctrl+Cなどのシャットダウンが起きたとき終了する

```
try:
    OutPut()
except rospy.ROSInterruptException:
    pass
```

OutPutVolte(Publisher)

- ROSのノードの設定

```
# RobotCarというトピックへString型をバッファ10で送信
pub = rospy.Publisher('RobotCar', String, queue_size = 10)
```

```
# ノード名をOutPutVolteで初期
rospy.init_node('OutPutVolte', anonymous = True)
```

```
# 1秒間に10回Publisherが動作
rate = rospy.Rate(10) # 10Hz
```

OutPutVolte(Publisher)

- 標準入力とメッセージ通信

```
while not rospy.is_shutdown(): # ノードが落ちない限り無限ループ
```

- 標準入力でモータの値を入力

```
left_value = raw_input() # 標準入力で左ボルト数を入力
```

```
right_value = raw_input() # 標準入力で右ボルト数を入力
```

- int型（整数型）に変換してメッセージに格納

```
msg.left_sval = int(left_value)
```

```
msg.right_sval = int(right_value)
```

OutPutVolte(Publisher)

- 標準入力とメッセージ通信

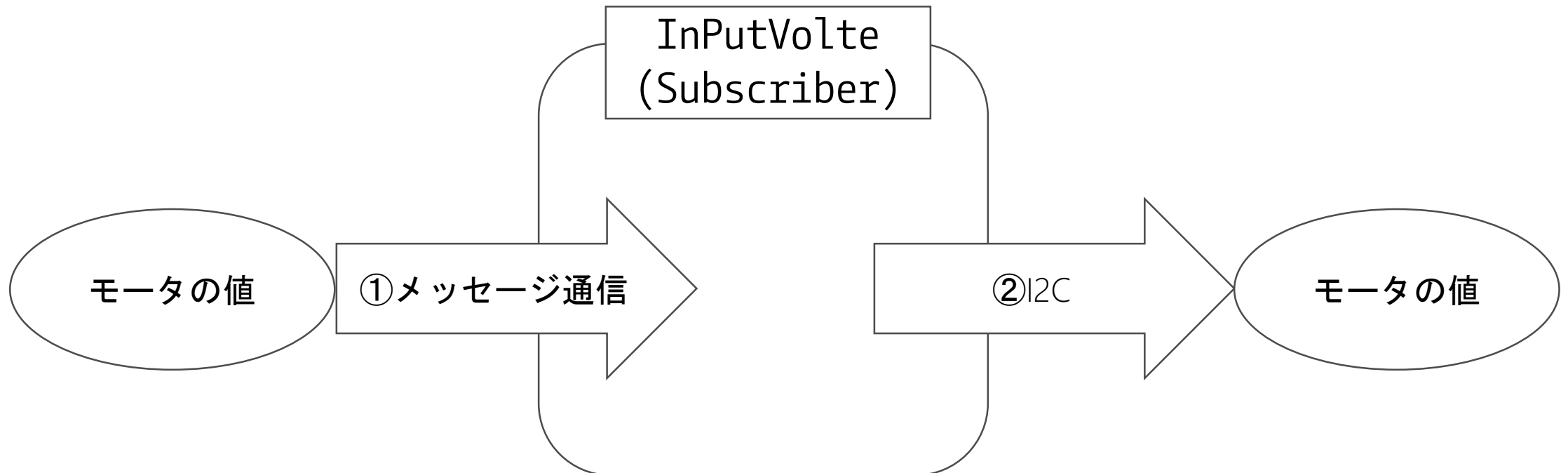
`pub.publish(msg)` # メッセージ通信で出力

`rate.sleep()` # プログラムをスリープさせる

InPutVolte(Subscriber)

• プログラム概要

- ① メッセージ通信でモータの値を入力
- ② I2C（シリアル通信）で値をデバイスに出力



InPutVolte(Subscriber)

- ROSのプログラミングに必要なimport
 - PythonでROSプログラムを作成
 - `import rospy # pythonでROSのプログラムを記述`
 - メッセージ通信で作成した型を使用する宣言
 - `from robotcar_pkg.msg import Msg # メッセージ通信で作成した型を使用`
- I2Cを使用するためにimport
 - `import smbus # I2Cを使用するためのモジュール`
- Sleep関数を使用するためにimport
 - `import time # sleepを使用するためのモジュール`

InPutVolte(Subscriber)

- モータドライバのアドレス

```
bus = smbus.SMBus(1)          # I2Cバス番号
```

```
SLAVE_ADDRESS_LEFT = 0x64    # 左モータのアドレス
```

```
SLAVE_ADDRESS_RIGHT = 0x63  # 右モータのアドレス
```

- 出力の設定

```
CONTROL = 0x00 # 出力の設定
```

```
STOP = 0x00    # モータ停止の値
```

InPutVolte(Subscriber)

- メインメソッド
 - プログラムの内容は以下のメソッドに書かれる

```
if __name__ == '__main__':
```

- ROSの処理を以下の関数で行う

```
InPut()
```

InPutVolte(Subscriber)

- InPut()

ノード名をInPutVolteで初期化

```
rospy.init_node('InPutVolte', anonymous = True)
```

RobotCarというトピックからString型受け取り,

callback関数で処理

```
rospy.Subscriber('RobotCar', String, callback)
```

ノードが止まるまで待機状態に移行する

```
rospy.spin()
```

InPutVolte(Subscriber)

- InPut()

rospy.spin()後に停止の値を入力

左モータ用DRV8830に停止の値を入力

bus.write_i2c_block_data(SLAVE_ADDRESS_LEFT, CONTROL, [STOP])

右モータ用DRV8830に停止の値を入力

bus.write_i2c_block_data(SLAVE_ADDRESS_RIGHT, CONTROL, [STOP])

InPutVolte(Subscriber)

- callback(msg):

```
# メッセージ通信から値を受け取る
```

```
left_sval = msg.left_sval
```

```
right_sval = msg.right_sval
```

```
print left_sval
```

```
print right_sval
```

```
# 左モータ用DRV8830に前進の値を入力
```

```
bus.write_i2c_block_data(SLAVE_ADDRESS_LEFT, CONTROL, [left_sval])
```

```
# 右モータ用DRV8830に前進の値を入力
```

```
bus.write_i2c_block_data(SLAVE_ADDRESS_RIGHT, CONTROL, [right_sval])
```

ROSのワークスペースをビルド4

- プログラムを作成したので，ワークスペースをビルドする

```
$ cd ~/catkin_ws/
```

```
$ catkin_make
```


プログラムの実行



roscore起動

- 現在使用しているTeraTerm上でroscoreを起動

```
$ cd ~/catkin_ws/
```

```
$ source /opt/ros/kinetic/setup.bash
```

```
$ source devel/setup.bash
```

```
$ roscore
```

```

raspberrypi.local:22 - roscore http://raspberrypi:11311/ VT
ファイル(E) 編集(E) 設定(S) コントロール(Q) ウィンドウ(W) ヘルプ(H)
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://raspberrypi:46687/
ros_comm version 1.12.14

SUMMARY
=====

PARAMETERS
* /rostdistro: kinetic
* /rosversion: 1.12.14

NODES

auto-starting new master
process[roscpp]: started with pid [818]
ROS_MASTER_URI=http://raspberrypi:11311/

setting /run_id to 7db0e138-e32b-11e8-a5c0-b827eb2a3af9
process[roscpp-1]: started with pid [831]
started core service [/roscpp]
  
```

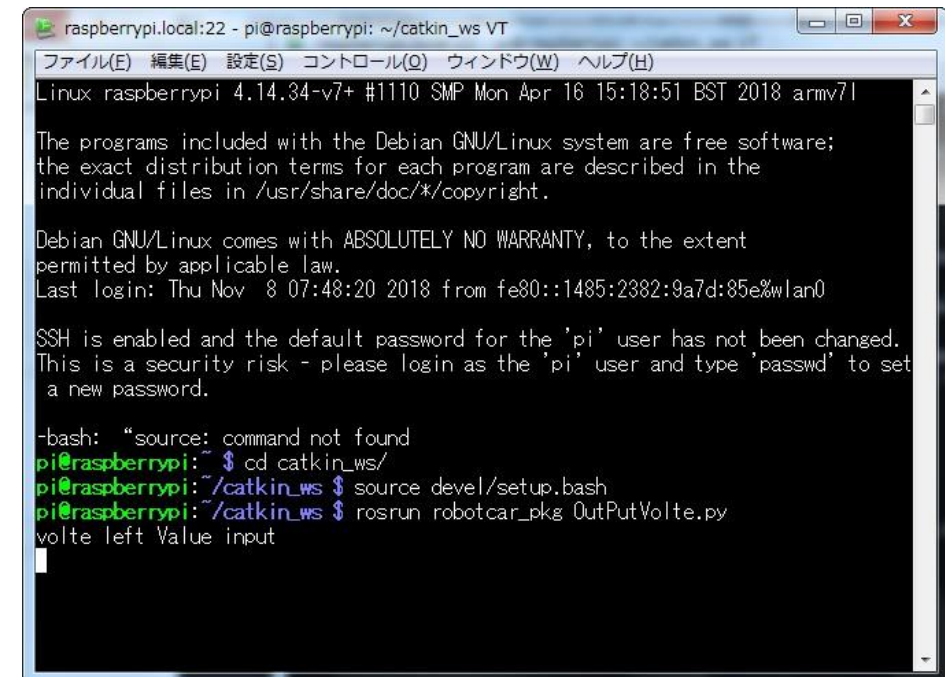
OutPutVolte起動

- OutPutVolteを起動するためにTeraTermを起動

```
$ cd catkin_ws/
```

```
$ source devel/setup.bash
```

```
$ rosrun robotcar_pkg OutPutVolte.py
```



```

raspberrypi.local:22 - pi@raspberrypi: ~/catkin_ws VT
ファイル(E) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
Linux raspberrypi 4.14.34-v7+ #1110 SMP Mon Apr 16 15:18:51 BST 2018 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov  8 07:48:20 2018 from fe80::1485:2382:9a7d:85e%wlan0

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

-bash: "source: command not found
pi@raspberrypi:~$ cd catkin_ws/
pi@raspberrypi:~/catkin_ws$ source devel/setup.bash
pi@raspberrypi:~/catkin_ws$ rosrun robotcar_pkg OutPutVolte.py
volte left Value input

```

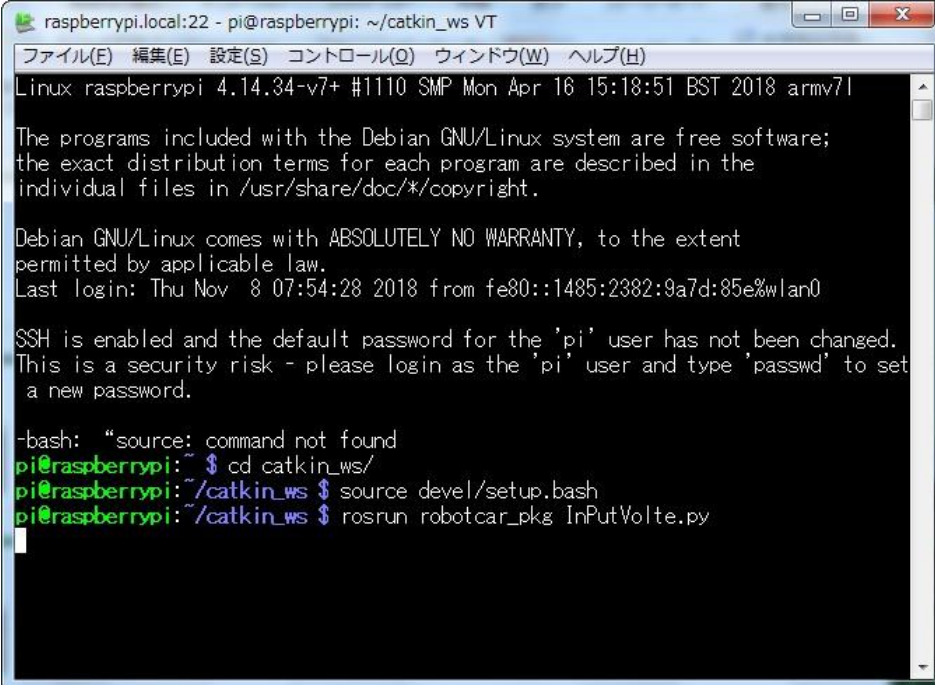
InPutVolte起動

- InPutVolteを起動するためにTeraTermを起動

```
$ cd catkin_ws/
```

```
$ source devel/setup.bash
```

```
$ rosrun robotcar_pkg InPutVolte.py
```



The screenshot shows a terminal window titled 'raspberrypi.local:22 - pi@raspberrypi: ~/catkin_ws VT'. The terminal output includes the Linux version (4.14.34-v7+), Debian GNU/Linux version (4.14.34-v7+), and the date (Mon Apr 16 15:18:51 BST 2018). It also displays the SSH warning and the execution of the 'source devel/setup.bash' and 'roslaunch robotcar_pkg InPutVolte.py' commands. The terminal text is as follows:

```

raspberrypi.local:22 - pi@raspberrypi: ~/catkin_ws VT
ファイル(E) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
Linux raspberrypi 4.14.34-v7+ #1110 SMP Mon Apr 16 15:18:51 BST 2018 armv7l
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov  8 07:54:28 2018 from fe80::1485:2382:9a7d:85e%wlan0
SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.
-bash: "source: command not found
pi@raspberrypi:~$ cd catkin_ws/
pi@raspberrypi:~/catkin_ws$ source devel/setup.bash
pi@raspberrypi:~/catkin_ws$ roslaunch robotcar_pkg InPutVolte.py

```

OutPutVolteからInPutVolteへ値を送る

- 標準入力で左右のモータの電圧値を入力

```

raspberrypi.local:22 - pi@raspberrypi: ~/catkin_ws VT
ファイル(E) 編集(E) 設定(S) コントロール(Q) ウィンドウ(W) ヘルプ(H)

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov  8 07:48:20 2018 from fe80::1485:2382:9a7d:85e%wlan0

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

-bash: "source: command not found
pi@raspberrypi:~ $ cd catkin_ws/
pi@raspberrypi:~/catkin_ws $ source devel/setup.bash
pi@raspberrypi:~/catkin_ws $ roslaunch robotcar_pkg OutPutVolte.py
volte left Value input
90
volte right Value input
90
volte Value send
volte left Value input

```

左右の値を90で入力

OutPutVolte

```

raspberrypi.local:22 - pi@raspberrypi: ~/catkin_ws VT
ファイル(E) 編集(E) 設定(S) コントロール(Q) ウィンドウ(W) ヘルプ(H)

Linux raspberrypi 4.14.34-v7+ #1110 SMP Mon Apr 16 15:18:51 BST 2018 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov  8 07:54:28 2018 from fe80::1485:2382:9a7d:85e%wlan0

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

-bash: "source: command not found
pi@raspberrypi:~ $ cd catkin_ws/
pi@raspberrypi:~/catkin_ws $ source devel/setup.bash
pi@raspberrypi:~/catkin_ws $ roslaunch robotcar_pkg InPutVolte.py
90
90

```

左右の値90が表示し、
RobotCarがバックする

InPutVolte

終了の仕方

- 各プログラム上で「**Ctrl+C**」で終了
 - OutPutVolteは標準入力の所で止まるので，適当な値を入力すること